Departamento de Ing. Matemática e Informática

Edificio las Encinas

Titulación:

INGENIERO TÉCNICO EN INFORMÁTICA DE GESTIÓN

Título del proyecto:

"HERRAMIENTAS DE AUTOR PARA DESARROLLO DE APLICACIONES MULTIMEDIA INTERACTIVAS MULTIUSUARIO: ANÁLISIS Y RECOMENDACIONES"

Alumno: Javier Jorge Soteras

Tutor: Alfredo Pina

Pamplona, 29 de Junio del 2012

Agradecimientos

A mi tutor Alfredo Pina por su ayuda y consejo siempre que lo he necesitado.

A mi padre, por preocuparse más que nadie.

A mi novia, por soportarme cuando nadie más lo hacía.

A mi familia, por haberme dado fuerzas.

A mis amigos y compañeros por su apoyo.

Índice

Capítulo 1. INTRODUCCIÓN....8
1.1. Herramientas para la programación autor....9
1.2. Componentes: Director y Flash....9
1.3. ¿Por qué Director y Flash?....10
Capítulo 2. PRIMEROS PASOS: MANEJO DE LAS HERRAMIENTAS AUTOR. .11
2.1. Interfaz de las Herramientas Autor:....11
2.2. Lenguaje de Autor:....18
Capítulo 3. USO DE LAS HERRAMIENTAS AUTOR: APLICACIÓN INTERACTIVA....20
Capítulo 4. MANEJO DE BASES DE DATOS SOBRE LAS HERRAMIENTAS AUTOR: 40
Capítulo 5. APLICACIONES MULTIUSUARIO 59
Capítulo 6. CONCLUSIONES....80
Capítulo 7. REFERENCIAS....81

Índice de figuras

Capítulo 1. INTRODUCCIÓN

La finalidad de este proyecto es la de orientar a desarrolladores de aplicaciones multimedia interactivas en el ámbito de las herramientas autor y ofrece recomendaciones a la hora de evolucionar de la herramienta Director a la herramienta Flash.

Para ello se analizarán las herramientas Director y Flash (con sus respectivos lenguajes de programación, Lingo y ActionScript). Este análisis será comparativo, ayudando en todo momento al usuario a comprender la transición de Director a Flash.

Finalmente se desarrollaran una serie de aplicaciones para ilustrar esta analítica.

Los puntos más importantes a analizar serán los siguientes:

- Análisis entre el distinto manejo de Director frente a Flash en las funciones de interactividad
- tratamiento de información multimedia,
- acceso a bases de datos, tanto locales como remotas,
- posibilidades de multiusuario (sincronizando mediante BD),

El PFC se plantea como un análisis sobre cómo construir una o varias aplicaciones que pueden ser desarrolladas sobre director y flash con propiedades anteriormente descritas. Junto a este análisis se mostrará al menos un ejemplo de aplicación con todos o la mayoría de los campos citados.

En un principio trataremos de resolver las diferencias entre Director y Flash sobre los distintos campos (Interactividad, uso de películas en la red, video, capacidad de multiusuario, uso de BD, uso de BD en la red) para después desarrollar una o varias aplicaciones que acrediten dicha funcionalidad.

El primer capítulo iniciará al desarrollador en las herramientas multimedia Adobe Flash CS3 y Macromedia Director MX 2004. A continuación, en el segundo capítulo se desarrollará una galería/reproductor para ilustrar el uso de las herramientas autor.

El tercer capítulo ayudará al usuario a comunicar las herramientas con una base de datos, y para ello, se creará una aplicación consola de comandos SQL.

Como último y cuarto capítulo, se englobarán todos los conocimientos adquiridos en el proceso e introducirá al desarrollador en aplicaciones multiusuario. Para ello, se creará un chat que mediante una BD sincronice la aplicación.

1.1. Herramientas para la programación autor

Antes que nada, deberíamos preguntarnos, ¿Qué es la programación autor? **La programación autor** es un lenguaje orientado al desarrollo de documentos multimedia.

Una herramienta de autor es aquella aplicación que se emplea para realizar el desarrollo del documento multimedia que se desea originar.

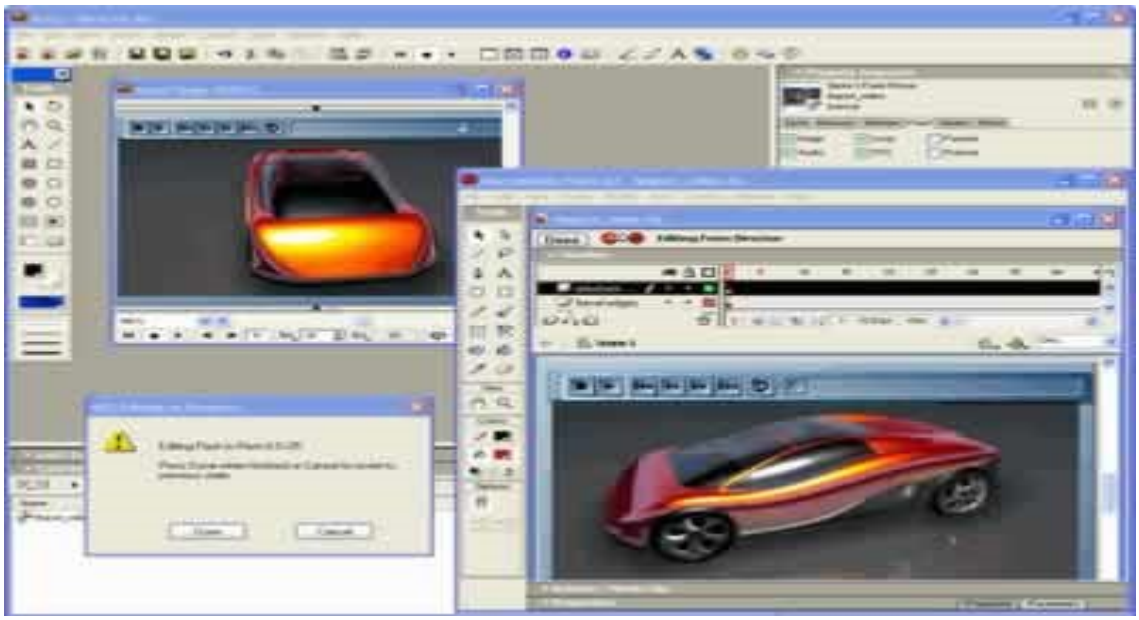


Figura 1.1 Herramientas Autor

1.2. Componentes: Director y Flash

Tanto Macromedia Director como Adobe Flash son herramientas de autor compuestas por un software que permite escribir texto y mezclarlo con audio, video, gráficos y animaciones, y con todo ese material construir una presentación que puede ser interactiva, una película interactiva.

Figura 1.2 Director y Flash

1.3. ¿Por qué Director y Flash?

Director nació en el año 1985 y evolucionó como una poderosa herramienta de integración de medios digitales, de alta calidad, y que también generó una arista para su incorporación a la Web, Shockwave. En cambio, Flash nació en 1996, orientado al desarrollo de aplicaciones multimedia exclusivo para la Web, y en poco tiempo evolucionó poderosamente.

Aunque Flash es actualmente el más extendido, popular y sobre el que más se desarrolla, Shockwave mantiene una fuerte posición por el número de ordenadores donde está instalado.

Otras características no incorporadas por Flash y sí por Shockwave, incluyen un motor de render mucho más rápido, junto con aceleración 3D por medio de hardware. Además, a través de los Xtra, los desarrolladores pueden ampliar la funcionalidad de Shockwave con aplicaciones hechas a medida. No todos son factores positivos, estas ventajas acarrean la desventaja de hacer las aplicaciones más pesadas en comparación con Flash. Este es uno de los factores que influyen en gran medida a la hora de volcar aplicaciones sobre la Web.

Macromedia Shockwave Player, el plugin necesario para la ejecución de las aplicaciones Director, está instalado en un 50% de los navegadores, frente a Flash Player, instalado en un 98% de los navegadores.

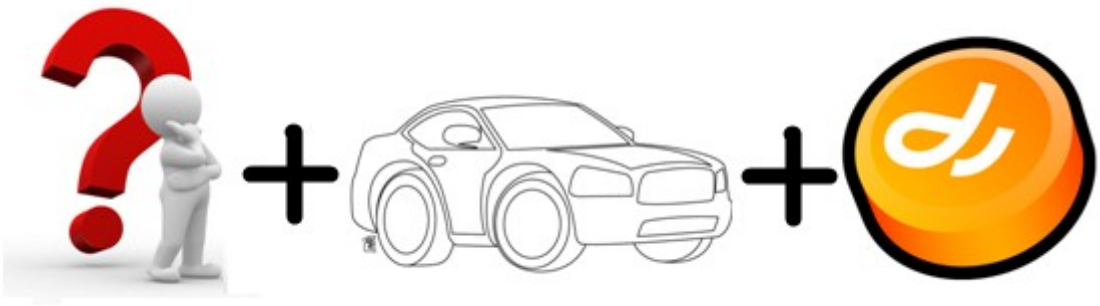


Figura 1.3 Por Qué Director y Flash

Capítulo 2. PRIMEROS PASOS: MANEJO DE LAS HERRAMIENTAS AUTOR

Como ya hemos explicado anteriormente, las herramientas que emplearemos serán Director y Flash, más exactamente Macromedia Director MX 2004 y Adobe Flash CS3.

Ya nos hemos introducido al concepto general de las herramientas autor y cuales son aquellas que emplearemos, ahora nos familiarizaremos con los componentes básicos de cada herramienta para lograr nuestra película final.

2.1. Interfaz de las Herramientas Autor:

2.1.1. Herramientas Macromedia Director MX 2004

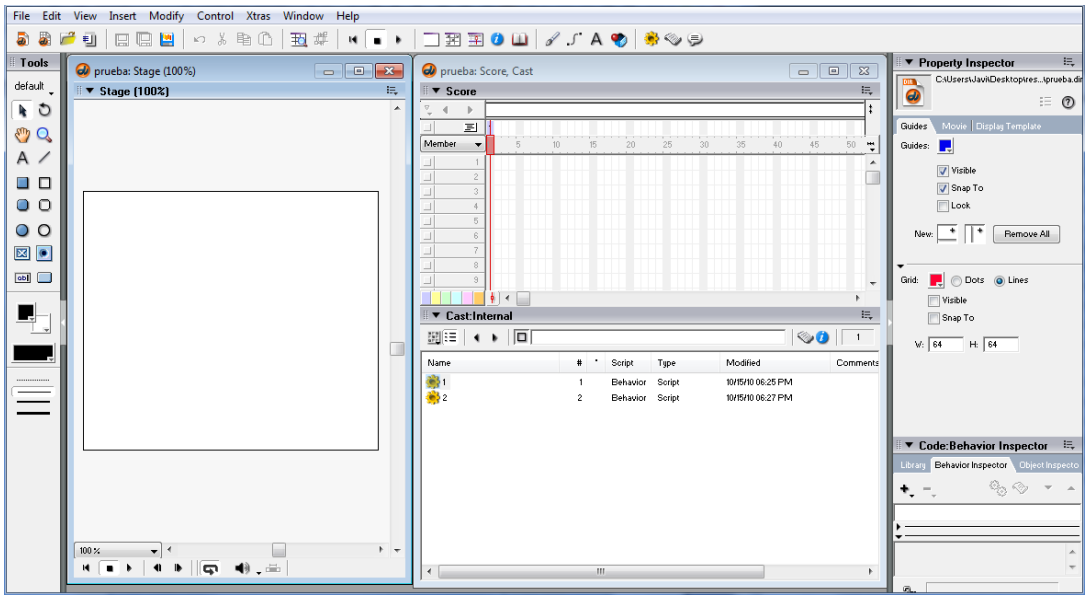


Figura 2.1.1. Macromedia Director MX 2004

- **La ventana Score:**

El Score se representa como una rejilla en la que las filas se denominan canales y las columnas se denominan fotogramas, mostrando en todo momento la posición de cada elemento y su relación con los demás en el Stage.

Cuando la cabeza de reproducción comience, leerá todo lo que encuentra, a la velocidad que le hayamos asignado, interpretando y obedeciendo a los scripts que encuentre en su camino y moviendo o mostrando los sprites que hayamos colocado en los canales.

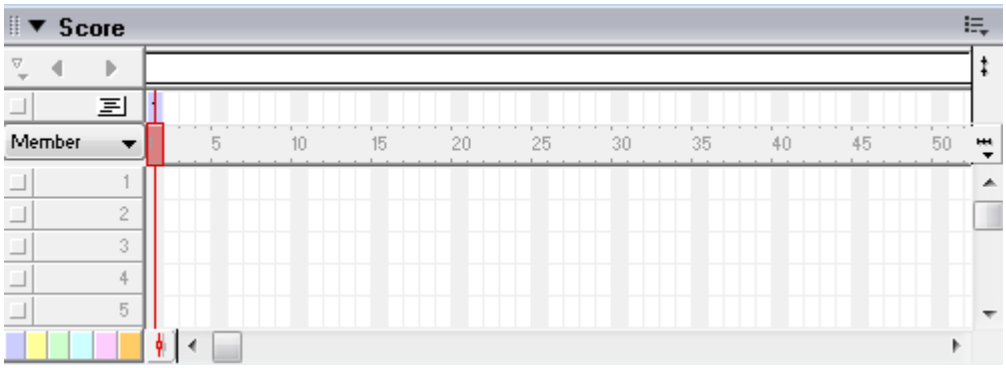


Figura 2.1.1.1. Herramienta Director: Ventana Score

- **La ventana Cast**:

La ventana Cast contiene los gráficos y el texto que se usarán en cada uno de los Sprites, así como los sonidos, el vídeo digital, las transiciones, los Scripts, y cualquier otro recurso que se utilice en Director. A cada uno de estos recursos se les denomina Cast Member o miembro de reparto.

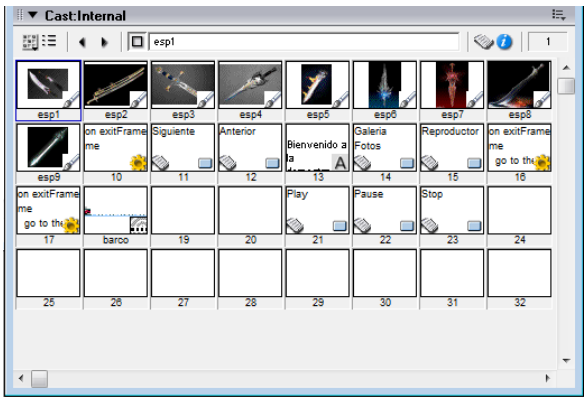


Figura 2.1.1.2. Herramienta Director: Ventana Cast

- **La ventana Stage**:

Esta ventana nos muestra el contenido de nuestra película, correspondiendo al fotograma en que nos encontremos en cada momento. En el momento que comience la reproducción de la película y hasta que esta se detenga, está ventana mostrará los resultados de nuestra aplicación.

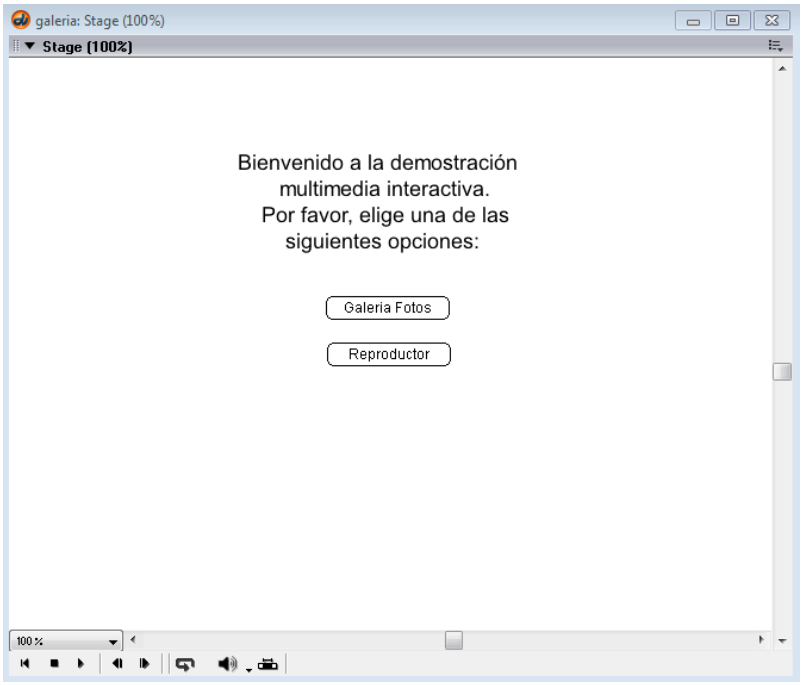


Figura 2.1.1.3. Herramienta Director: Componente Stage

- **La barra de Herramientas**:

Las herramientas más comunes que necesitaremos se localizan en esta sección, accesible en cualquier momento. No obstante, en caso de necesitarlas, existen más herramientas de las que se visualizan inicialmente.

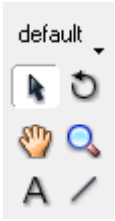


Figura 2.1.1.4. Herramienta Director: Barra de Herramientas

- **Los Inspectores:**

Los inspectores se utilizan para ver y modificar los atributos de los miembros de reparto de texto, sprites y behaviors.

Existen cuatro tipos de inspectores y cada uno realiza una tarea específica:

- **Property Inspector**: muestra las propiedades actuales de cualquier elemento seleccionado, incluyendo el *stage*, así como una película activa. Pueden ajustarse cualquiera de las propiedades disponibles de casi todos los elementos de la película con sólo utilizar esta ventana.

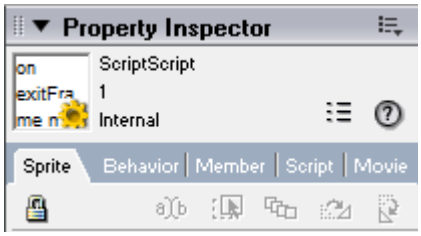


Figura 2.1.1.5. Herramienta Director: Property Inspector

- **Behavior Inspector**: se usa para crear nuevos *behaviors* (comportamientos, ó scripts de Lingo) y para modificar los existentes.

Figura 2.1.1.6. Herramienta Director: Behavior Inspector

- **Text Inspector**: se usa para dar formato al texto y para añadir hipervínculos al mismo.

Figura 2.1.1.7. Herramienta Director: Text Inspector

- **Memory Inspector**: muestra la cantidad de memoria RAM que está siendo utilizada por la aplicación de Director y los elemntos que contiene una película.

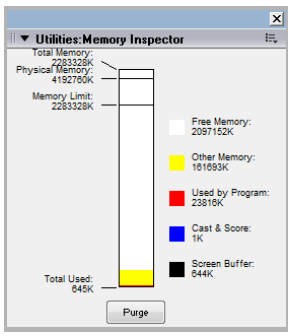


Figura 2.1.1.8. Herramienta Director: Memory Inspector

2.1.2. **Herramientas Adobe Flash CS3**

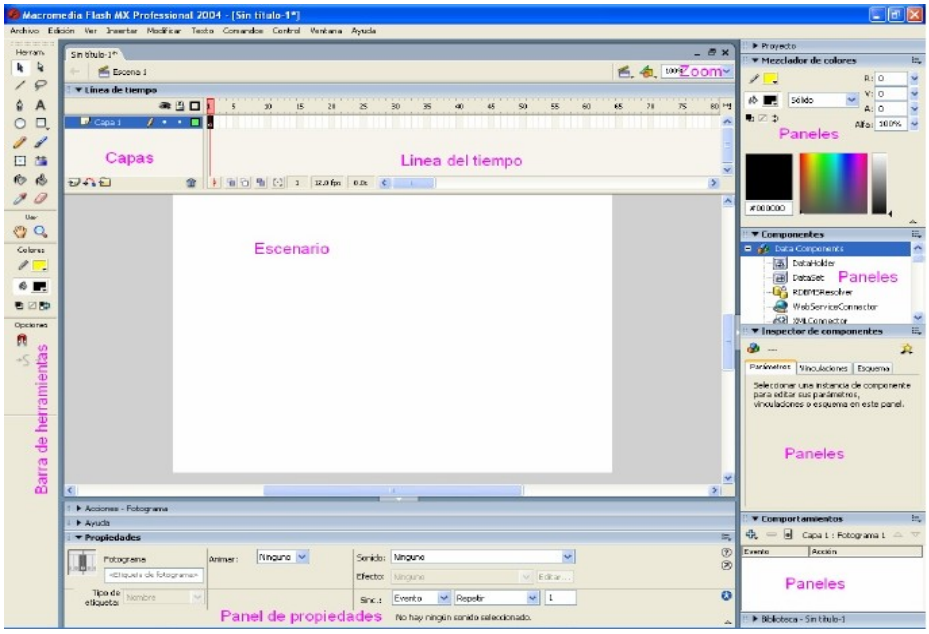


Figura 2.1.2. Adobe Flash CS3

- **La línea de tiempo:**

Es uno de los elementos indispensables a la hora de crear objetos que se modifiquen con el paso del tiempo. A nivel conceptual, la línea de tiempo representa la sucesión de fotogramas en el tiempo. Es decir, la película Flash no será nada más que los fotogramas que aparecen en la línea de tiempo uno detrás de otro, en el orden que establece la misma línea de tiempo. Representa la ventana Store en Director:

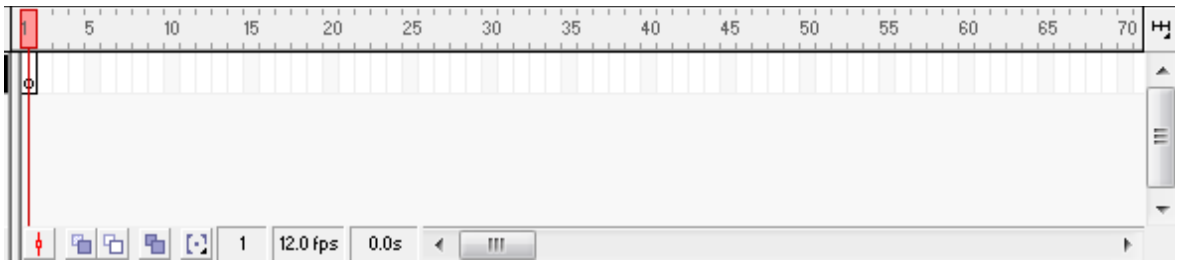


Figura 2.1.2.1. Herramientas Flash: Línea de tiempo

Consta de dos partes:

- **Los Fotogramas (frames)** que vienen delimitados por líneas verticales (formando rectángulos).
- **Los Números de Fotograma** que permiten saber qué número tiene asignado cada fotograma, cuánto dura o cuándo aparecerá en la película.

Además, en la parte inferior hay que proporcionan mayor información sobre el número de fotograma actual, la velocidad de los fotogramas y el tiempo de película transcurrido.

- **Las Capas**

El concepto de capa es fundamental para manejar Flash de forma eficiente. Una Capa se puede definir como una película independiente de un único nivel. Es decir, una capa contiene su propia Línea de Tiempo (con infinitos fotogramas). Representan a los Sprites en Director:

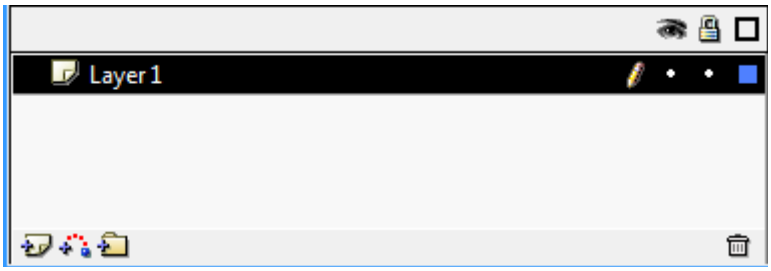


Figura 2.1.2.2. Herramientas Flash: Capas

Los objetos que estén en una determinada capa comparten fotograma y por tanto, pueden "mezclarse" entre sí. Veamos un ejemplo

Paso 1: Creamos un cuadrado:

Figura 2.1.2.3. Herramientas Flash: Capas, elemento 1

Paso2: En la misma capa creamos un círculo encima del cuadrado:

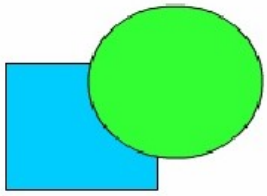


Figura 2.1.2.4. Herramientas Flash: Capas, elemento 2

Paso 3: Ahora movemos el círculo a ver que pasa:

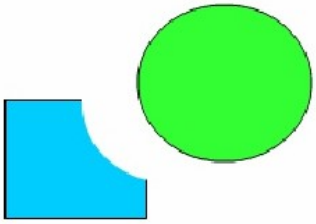


Figura 2.1.2.5. Herramientas Flash: Capas, elemento 3

Para que unos objetos no interfieran en otros objetos, los colocamos en capas diferentes. Podemos crear tantas capas como nos sea necesario.

- **La ventana Scene**

Esta ventana nos muestra el contenido de nuestra película flash, correspondiendo al fotograma en que nos encontremos en cada momento. En el momento que comience la reproducción de la película y hasta que esta se detenga, está ventana mostrará los resultados de nuestra aplicación. Representa la ventana Stage en Director.

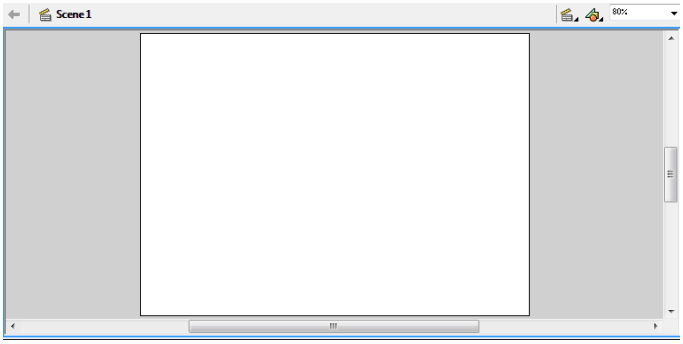


Figura 2.1.2.6. Herramientas Flash: Ventana Scene

- **La barra de Herramientas**:

Las herramientas más comunes que necesitaremos se localizan en este panel, accesible en cualquier momento. No obstante, en caso de necesitarlas, existen más herramientas de las que se visualizan inicialmente.

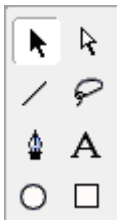


Figura 2.1.2.7. Herramientas Flash: Barra de Herramientas

- **Los Paneles**

Los Paneles son conjuntos de comandos agrupados según su función (por ejemplo, todo lo que haga referencia a los colores, irá en el Panel "Mezclador de Colores"). Su misión es simplificar y facilitar el uso de los comandos. Representan a los inspectores en Director:

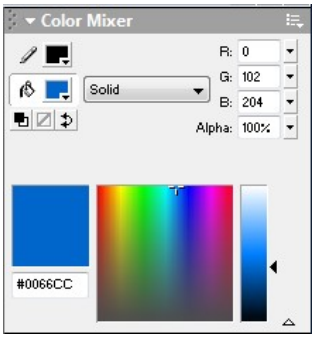


Figura 2.1.2.8. Herramientas Flash: Paneles

- **La biblioteca**

En el caso de Flash, cuando importemos un objeto, este se verá reflejado en la biblioteca. Si lo comparamos con director, tiene una funcionalidad parecida a la ventana Cast, ya que cada vez que añadamos un objeto podemos crear infinitas instancias del mismo y si el objeto contiene alguna propiedad a nivel de objeto, todas las instancias de este también las compartirán.

Figura 2.1.2.9. Herramientas Flash: Biblioteca

2.2. Lenguaje de Autor:

El lenguaje Autor es un lenguaje orientado al desarrollo de documentos multimedia. En este proyecto emplearemos Lingo, para Macromedia Director, y Action Script, para Macromedia Flash.

El objeto de este proyecto no es el de explicar el comportamiento de ambos lenguajes, por lo que se hará una breve mención de ambos.

2.2.1. Lingo para Macromedia Director MX 2004

Lingo es un lenguaje interpretado. El código se escribe en *scripts* que son ejecutados secuencialmente.

Sirve para asociar controles a eventos, condiciones en animaciones de Director y realizar cálculos como cualquier otro lenguaje.

Los tipos de Scripts pueden ser variados:

1- Behaviors: Asociados a *sprites* o *frames*. Un script puede estar asociado a varios elementos (múltiples *sprites o frames*).
2- Movies: Asociados a toda la animación, por ejemplo: stop, pausa.

3- Scripts asociados a *cast*: Cada vez que creamos un *sprite* tendrá asociado el script.

Como ejemplo, un script asociado a un frame:

```
1 on exitFrame me
2   go to the frame
3 end
```

Figura 2.2.1.1. Ejemplo Lingo

Con este código, nos dirigiremos al comienzo del *frame* sobre el que se ejecuta este script, una vez halla llegado al final del *frame* del mismo.

2.2.2. Action Script para Adobe Flash CS3

ActionScript es el lenguaje de programación para crear scripts en Flash. Estos scripts pueden asociar acciones a fotogramas u objetos.

1- Acciones de Fotograma: Cuando la cabecera de reproducción de la línea de tiempo pase sobre el fotograma clave que contiene el script, las instrucciones añadidas al fotograma clave se ejecutarán
2- Acciones de Objeto: Cuando se reproduzca el evento en el objeto sobre el que hemos generado el script, las instrucciones generadas al evento del objeto se ejecutarán.

Como ejemplo, una acción asociada a un objeto:

```
on (press) {
    gotoAndPlay (60);
}
```

Figura 2.2.2.1. Ejemplo ActionScript

Este script está asociado al evento *press* de un botón, y cuando este sea ejecutado, nos dirigirá al fotograma 60.

Capítulo 3. USO DE LAS HERRAMIENTAS AUTOR: APLICACIÓN INTERACTIVA

Para ilustrar correctamente el uso de las herramientas Autor, crearemos una sencilla galería en la que recorreremos una serie de fotografías o bien reproduciremos una película.

Para ello crearemos una sencilla interfaz en la que añadiremos botones de control para manejarnos por las distintas fotografías de la galería y un botón para cambiar de galería a reproductor y reproducir la película.

3.1. Galería sobre Macromedia Director MX 2004

- **Los miembros del reparto (cast members)**:

Lo primero que haremos será importar nuestros miembros del reparto (fotografías en este caso) que serán las que más adelante se ilustrarán en la galería. Para ello seleccionar la pestaña File → Import… y escoger el elemento que queremos importar al proyecto.

Una vez realizado este proceso con todos los archivos que necesitamos, podemos emplear la ventana Cast para renombrar los elementos importados.

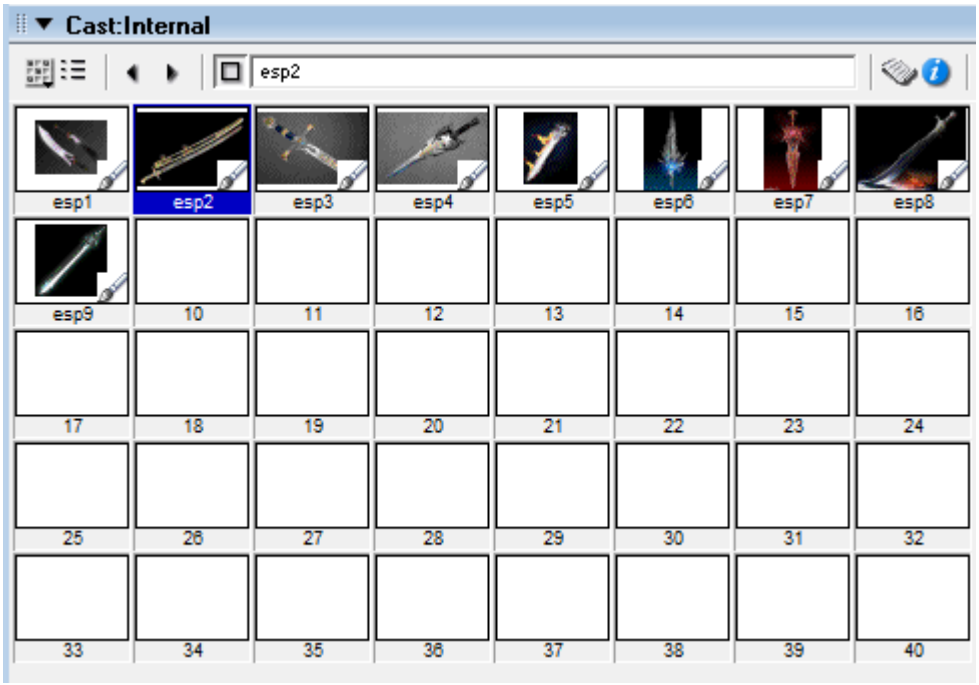


Figura 3.1.1. Galería Director: Miembros del Reparto

El siguiente paso será crear dos botones para poder navegar por la galería. Para ello será suficiente con Insert → Control → Push Button

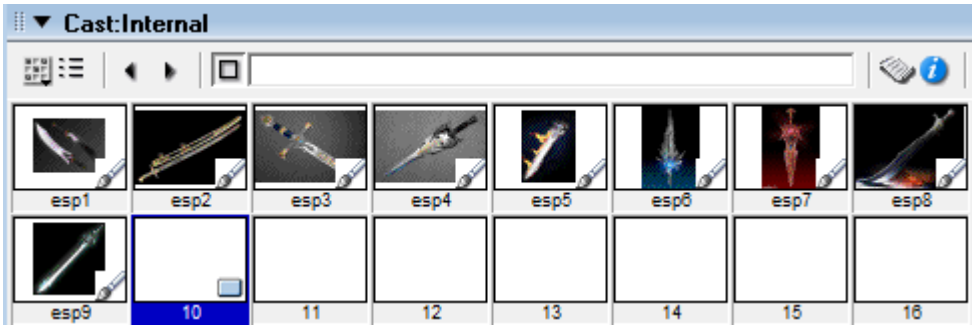


Figura 3.1.2. Galería Director: Miembros del Reparto

Para editar el texto que contendrá el botón, simplemente seleccionamos el cast member con doble clic y en la parte izquierda especificaremos el texto del botón:

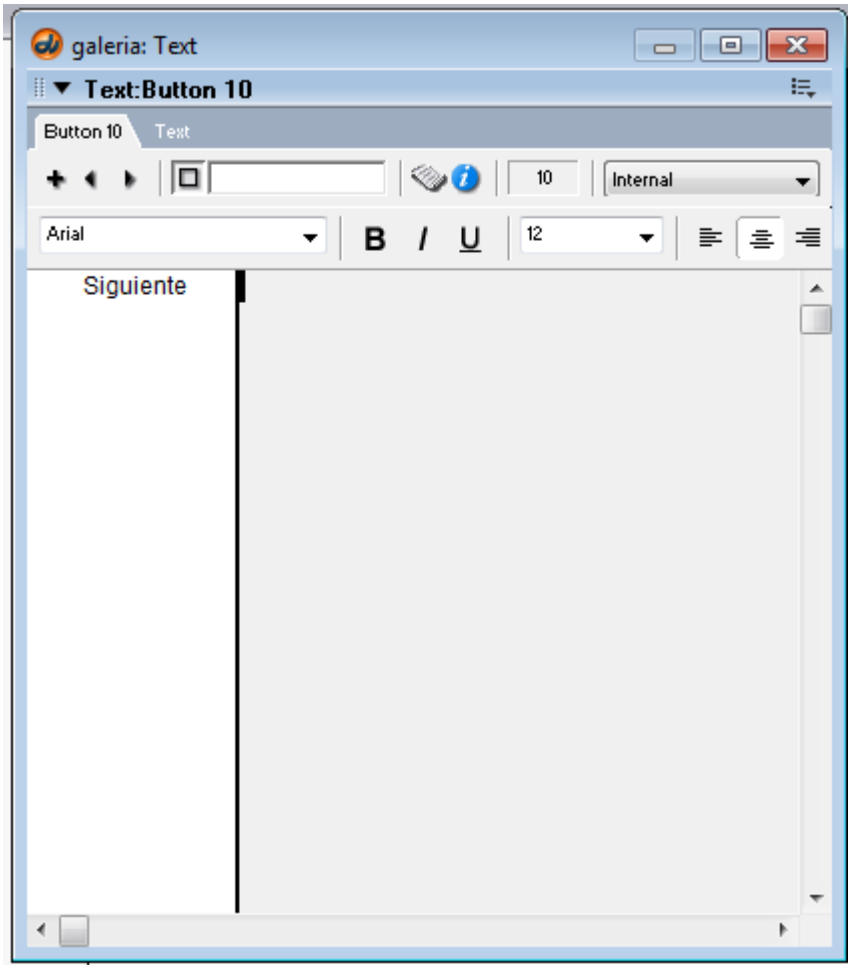


Figura 3.1.3. Galería Director: Contenido de un Miembro del Reparto

Del mismo modo añadiremos los botones de selector de galería y reproductor, botones de play, pause y stop para el reproductor, un gif que contendrá el elemento a reproducir y un elemento de texto que contendrá una breve explicación.

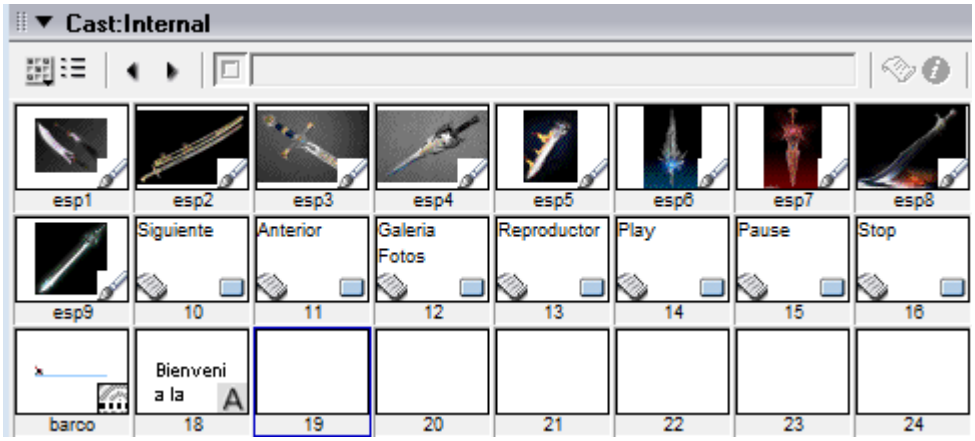


Figura 3.1.4. Galería Director: Miembros del Reparto

- **Configurando la ventana Score:**

Como hemos explicado antes, la ventana Score representa la vida de la película en una sucesión de fotogramas. Cada vez que avancemos por un fotograma, la película

actuará en consecuencia y cambiará sus contenidos en función del fotograma en el que estemos localizados.

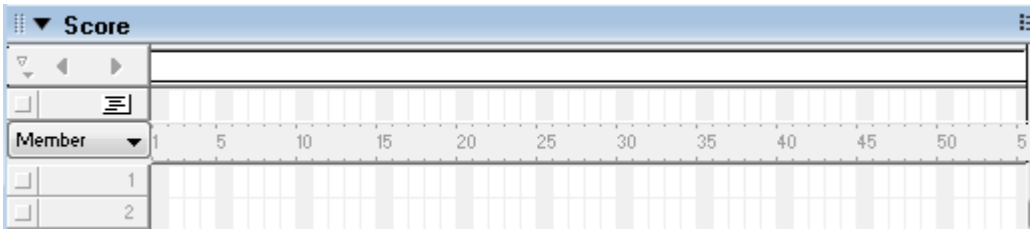


Figura 3.1.5. Galería Director: Ventana Score

Para nuestra galería emplearemos únicamente 3 fotogramas: Un fotograma como menú, un fotograma para la galería de fotos y un fotograma para el reproductor de la película.

Por defecto, Director recorre 24 fotogramas por segundo, por lo que si únicamente queremos emplear 3 fotogramas, deberemos crear un script por fotograma, para mantenernos en el mismo fotograma cada vez que este termine su reproducción.

Para ello crearemos 3 scripts asociados cada uno a un fotograma, estos scripts serán de tipo Behaviors. Seleccionaremos los fotogramas uno a uno y o bien haciendo doble clic sobre cada uno de estos, o bien seleccionando un nuevo Behavior en la ventana de Behavior Inspector una vez seleccionado el fotograma, se abrirá una ventana en la que podremos editar contenido en lenguaje Lingo.

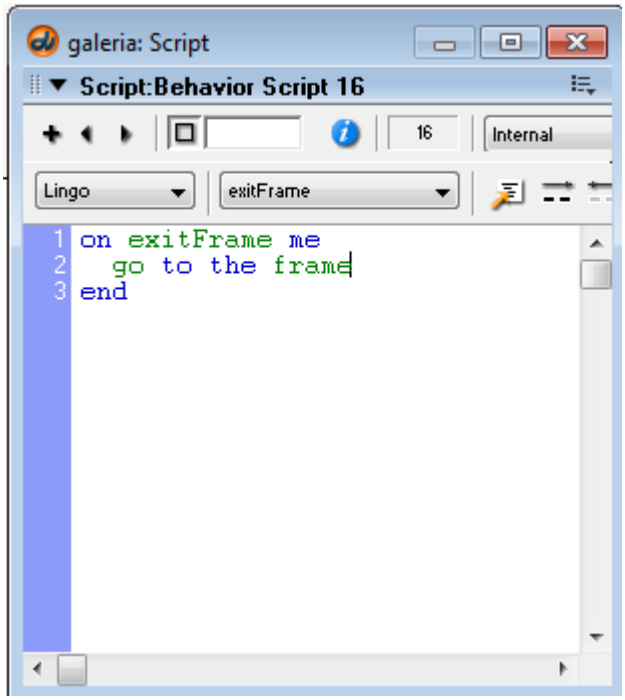


Figura 3.1.6. Galería Director: Script a nivel de Fotograma

La ventana de Score ahora contendrá estos tres scripts que hemos definido, por lo que, por el momento, la película no tendrá fin.

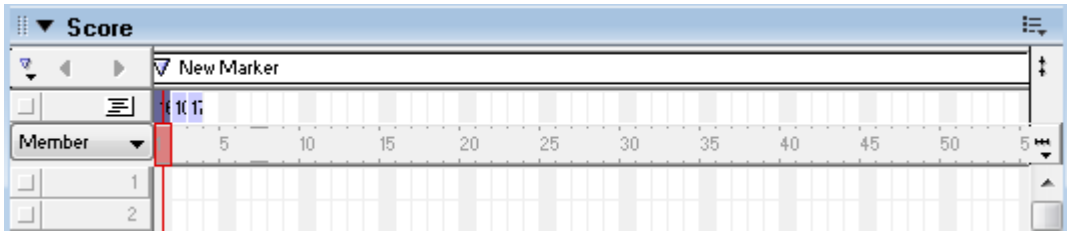


Figura 3.1.7. Galería Director: Ventana Score con Script Behaviors

Podemos ver que estos nuevos scripts se ven reflejados en la ventana Cast como nuevas entradas de Cast Members.

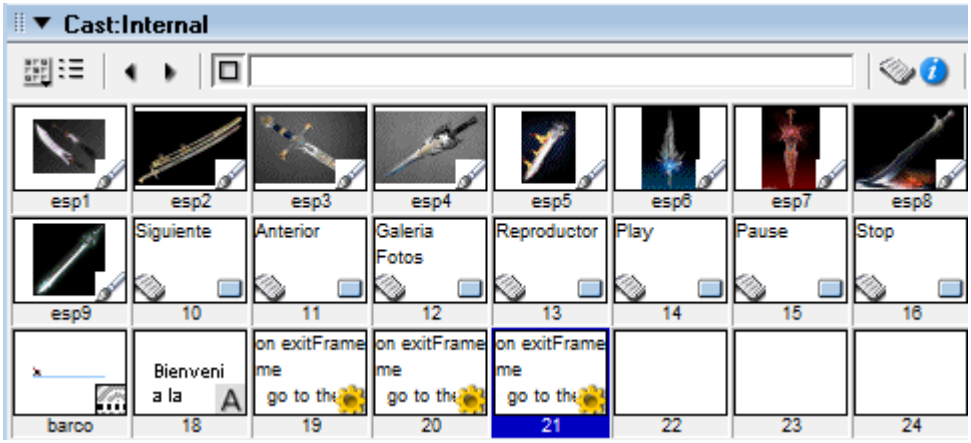


Figura 3.1.8. Galería Director: Miembros del Reparto

Ahora que la película está compuesta por tres fotogramas, el siguiente paso es darle contenido a cada uno de los fotogramas.

- **Contenido del fotograma 1: Presentación**

En este fotograma presentaremos una pantalla inicial en la que el usuario podrá navegar en dirección a la galería de fotos o al reproductor. Para ello seleccionaremos el fotograma 1 de la ventana Cast.

El siguiente paso será arrastrar los Cast Members "Texto de Bienvenida", "Galería de Fotos" y "Reproductor" a la ventana Stage, en la ubicación que queramos que ocupen. Una vez añadidos los elementos, estos se verán reflejados en la ventana Cast y deberemos reducir el intervalo de cada uno de los Sprites a 1 fotograma.

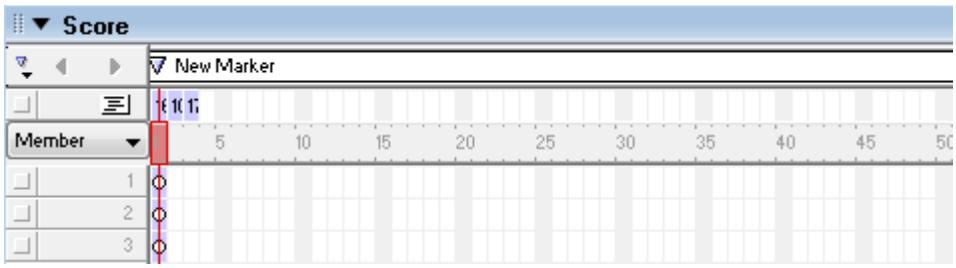


Figura 3.1.9. Galería Director: Ventana Cast con Sprites

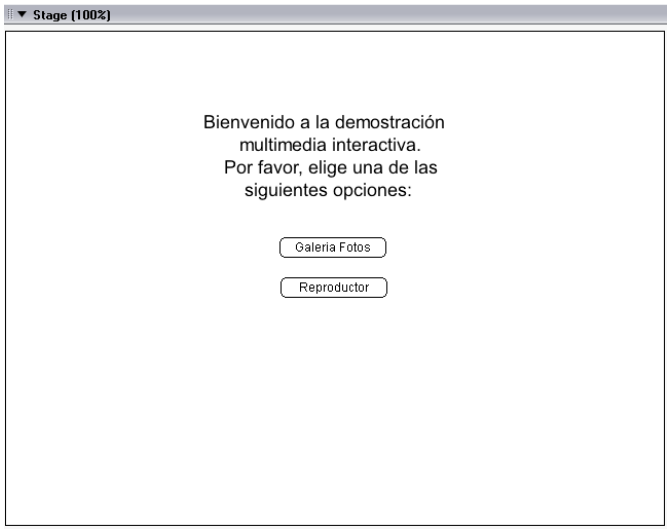


Figura 3.1.10. Galería Director: Ventana Stage: Fotograma 1

Por último, se les asignará a los Cast Member "Galería de Fotos" y "Reproductor" los scripts de navegación propios para cada uno de ellos. Al asignarles el script a nivel de Cast Members y no a nivel de Sprite, nos aseguramos que siempre que empleemos estos botones en algún momento de la película, estos contendrán las acciones que les hemos asociado.

Para "Galería de Fotos":

```
on mouseUp
  go to frame(2)
end
```

Para "Reproductor":

```
on mouseUp
  go to frame(3)
end
```

- **Contenido del fotograma 2: Galería de Fotos**

En este fotograma ilustraremos un menú de navegación y se tendrán que visualizar las fotos en la ventana Stage en función de la posición de la imagen central en el navegador. También tendremos la opción de partir al reproductor. Básicamente mostraremos todas las fotos simultáneamente, y en función de la posición del navegador, ocultaremos las imágenes que no se tengan que mostrar en ese momento.

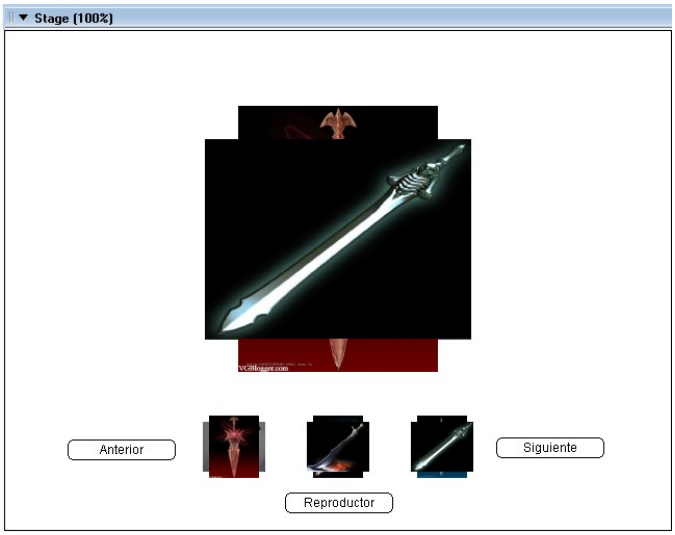


Figura 3.1.11. Galería Director: Ventana Stage: Fotograma 2 sin scrips

A continuación, a excepción de los tras primeras imágenes (Sprites) de navegación, seleccionaremos todos los Sprites de imágenes y en la ventana Property Inspector, a nivel de Sprite, en la propiedad Ink: copy, seleccionaremos una opacidad de un 0%, para que las imágenes queden completamente invisibles. A los Cast Member "Anterior" y "Siguiente" les añadiremos los scripts que nos permitirán la navegación por la galería.

Para que estos scripts resulten, deberemos haber añadido en orden los Sprite, primero las 9 imágenes en miniatura, después las 9 imágenes en formato de visualización y después los botones de navegación.

Nos basaremos en el Sprite que se situa en medio del panel de navegación para visualizar el Sprite protagonista que representará la imagen a mostrar. Siempre que accionemos uno de los botones "Anterior" y/o "Siguiente", aumentaremos o disminuiremos en uno el Sprite que queremos visualizar.

Para el Cast Member "Anterior" el código script que deberá tener es el siguiente:

```
on mouseUp
  if sprite(1).blend <> 100 then
    repeat with z = 1 to 9
      x = sprite(z).locH
      x = x + 100
      sprite(z).locH = x
      if x > 420 then
        x = 220
        sprite(z).blend = 0
      end if
      if sprite(z).blend = 0 and z>3 then
        if sprite(z-1).blend = 100 and sprite(z-2).blend =
100 then
          sprite(z-3).blend = 100
        end if
      end if
      sprite(z).locH = x
    end repeat
  end if
end
```

Y para el Cast Member "Siguiente":

```
on mouseUp
  if sprite(9).blend <> 100 then
    repeat with z = 1 to 9
      x = sprite(10-z).locH
      x = x - 100
      sprite(10-z).locH = x
      if x < 220 then
        x = 420
        sprite(10-z).blend = 0
      end if
```

```
        if sprite(10-z).blend = 0 and sprite(10-z+1).blend =
100 and sprite(10-z+2).blend = 100 then
          sprite(10-z+3).blend = 100
        end if
        sprite(10-z).locH = x
      end repeat
    end if
end
```

Vemos como irá desplazando los Sprite en función del Sprite que se esté visualizando en cada momento.

Para el Sprite "Reproductor" no deberemos añadirle ningún script, ya que este tiene asociado uno a nivel de Cast Member.

El fotograma resultante:

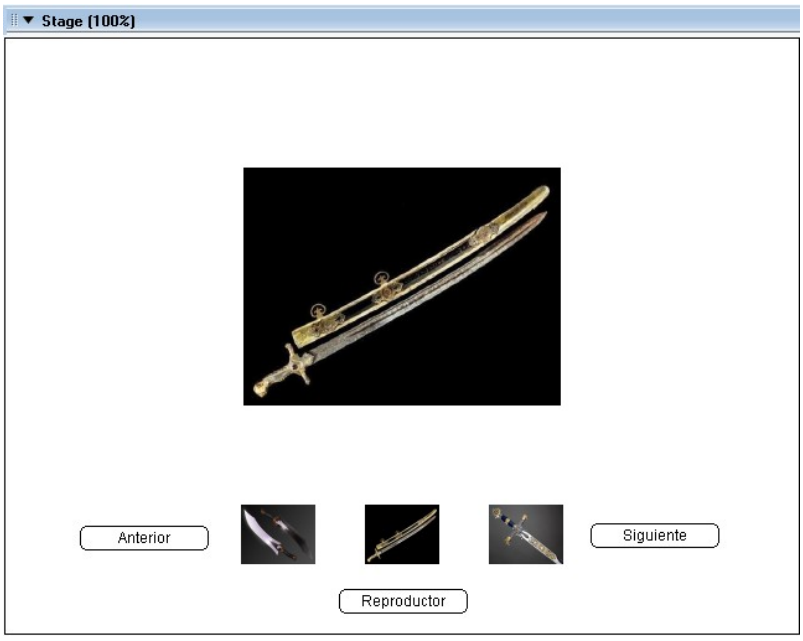


Figura 3.1.12. Galería Director: Ventana Stage: Fotograma 2 final

- **Contenido del fotograma 3: Reproductor**

En este último fotograma visualizaremos una pequeña película, un GIF en este caso, pero podríamos hacer lo mismo con una película distinta.

Los nuevos Sprites que emplearemos serán los Cast Members de "Play", "Pause" y "Stop" para la reproducción de la película.

Lo primero será añadir los Sprites a la ventana Stage:

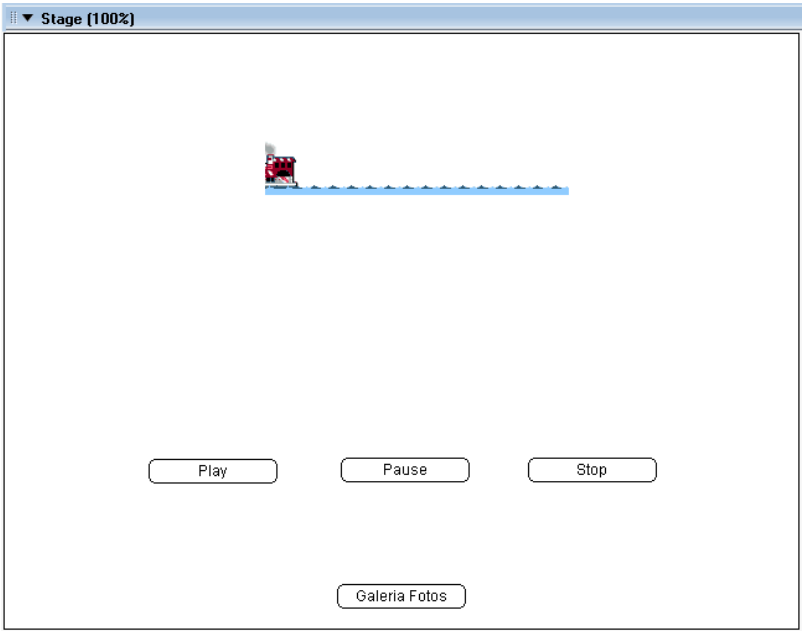


Figura 3.1.13. Galería Director: Ventana Stage: Fotograma 3

Los códigos que emplearemos mara manejar la película son:

Para "Play", lo único que haremos es acudir al fotograma de la película, ya que cuando estamos en ella, la película se reproduce automáticamente.

```
1 on mouseUp
2   go to frame(3)
3 end
```

Para "Pause" emplearemos el evento pause(), el cual paraliza la película en el punto en el que nos encontramos:

```
1 on mouseUp
2   pause()
3 end
```

Por último, para "Stop" refrescaremos la ventana Stage para que ningún pixel se quede donde no corresponda, resetearemos la película y llamaremos al evento pause() para que no comience sin que se lo ordenemos:

```
1 on mouseUp
2   updateStage
3   sprite(1).rewind()
4   go to frame(3)
5   pause()
6 end
```

- **Generando la película**

Para obtener el ejecutable con nuestra galería de fotos lo único de deberemos hacer es acceder a la pestaña File → Publish. Automáticamente se generará nuestra película en la ubicación sobre la que estemos trabajando.

3.2. Galería sobre Adobe Flash CS3

En el caso de la creación de la galería en Flash, intentaremos reproducir los pasos que recorrimos cuando creamos la galería en Director, para comparar paso a paso la construcción de cada uno.

- **Añadiendo elementos a la galeria**:

Lo primero que haremos será importar las imágenes que emplearemos para la galería. Para ello seleccionar la pestaña Archivo → Importar → Importar a Escenario.

Una vez realizado este proceso con todos los archivos que necesitamos, podemos emplear la biblioteca para renombrar los elementos importados.

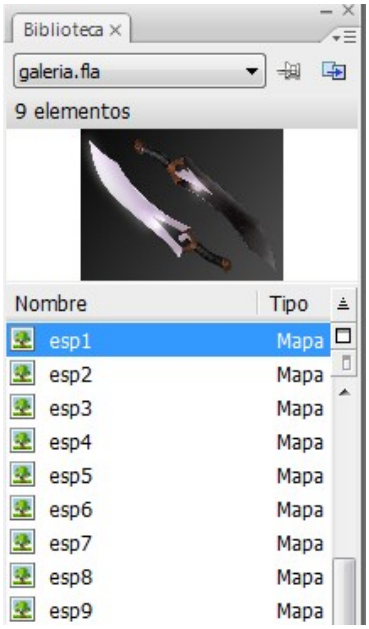


Figura 3.2.1. Galería Flash: Imágenes a Ilustrar

El siguiente paso será crear dos botones para poder navegar por la galería. Para ello seleccionaremos la pestaña Insertar → Nuevo Símbolo… → Botón

En el caso de Flash, los botones importados no contienen un contenido por defecto, por lo que debemos crearlos desde el principio.

Para darle forma al botón que acabamos de crear, lo primero que deberemos hacer es seleccionarlo en la biblioteca con doble Clic. Se mostrará la línea del tiempo del botón una vez que realizada dicha acción.

Un botón se compone de 4 fotogramas, aunque nosotros emplearemos únicamente 3 de ellos:

1. **Fotograma Reposo**: Es el estado inicial en el que se encuentra el botón.
2. **Fotograma Sobre**: Indica que cuando se pasa el cursor sobre el botón, este cambiará de posición o color según haya sido diseñado.

3. **Fotograma Presionado**: Incica que cuando se presione el botón, este cambiará también de color o de posición.

Estas tras acciones solo son visualizadas cuando el símbolo botón es puesto en la escena y es publicado.

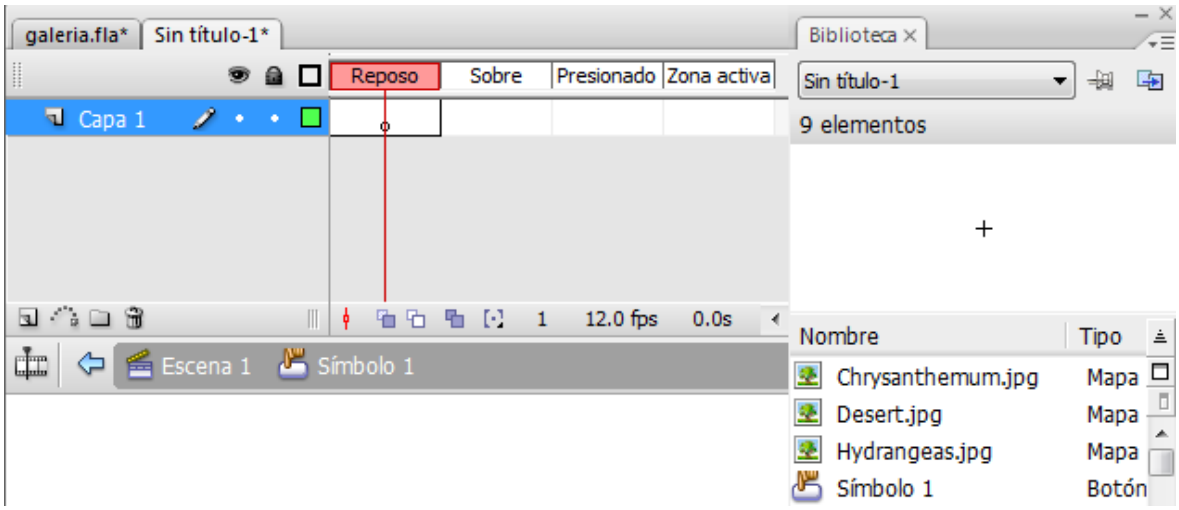


Figura 3.2.2. Galería Flash: Crear botón

A continuación se diseñará un botón con borde redondeado en el símbolo creado. Presionando en la herramienta rectángulo, luego seleccionando la herramienta de establecer Radio de esquina, en el panel emergente de configuración de rectángulo, se escribe 10 como Radio de esquina. Colocaremos el rectángulo en las coordenadas XY (0,0) y con tamaño de 100px de anchura y 20 de altura con la ayuda del panel de propiedades. Además, le daremos un grosor de 2 pixeles al borde del botón.

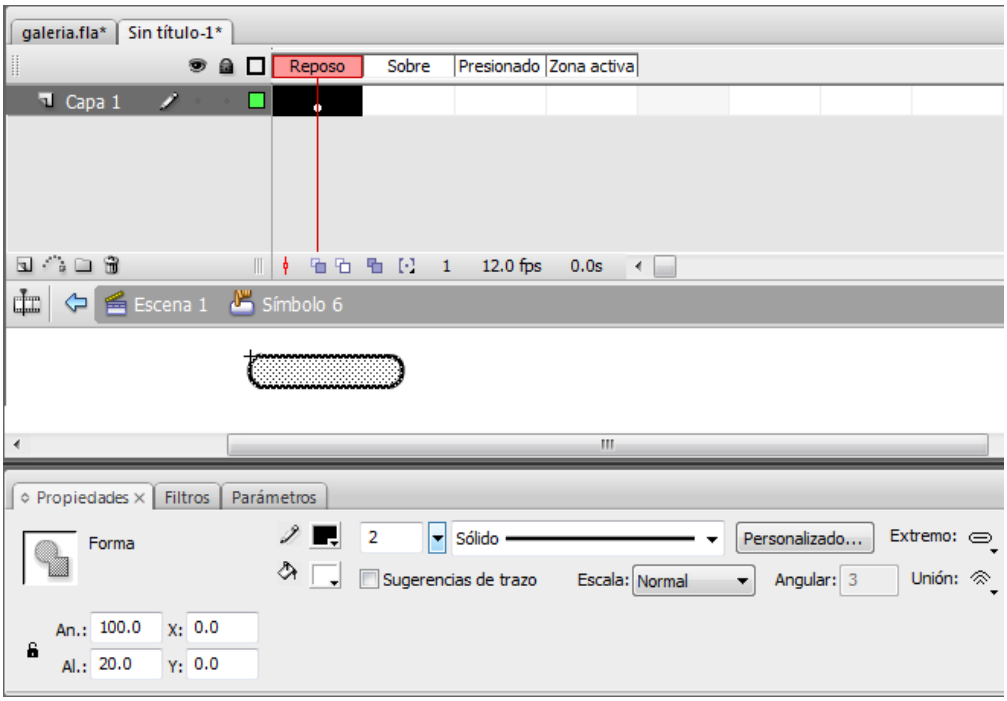


Figura 3.2.3. Galería Flash: Crear botón, Reposo

Para continuar con el siguiente fotograma, insertaremos un nuevo fotograma clave en el segundo estado de la línea del tiempo Sobre. Para ello seleccionar la pestaña de Insertar → Línea del tiempo → Fotograma clave o bien pulsar F6.

En este segundo estado no haremos ninguna modificación sobre el botón, ya que en director, los botones por defecto no realizan ninguna acción cuando situamos el cursero del ratón encima de alguno de estos.

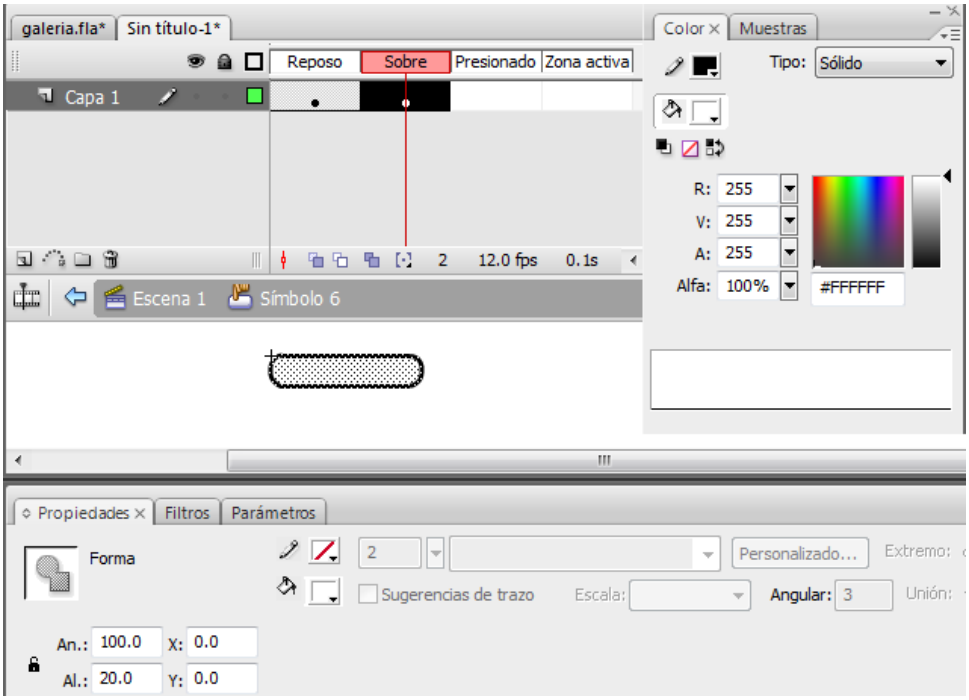


Figura 3.2.4. Galería Flash: Crear botón, Sobre

Por último, volveremos a añadir un fotograma clave para el estado Presionado. En este caso haremos que el botón obtenga el mismo color que los botones de Director cuando estos son pulsados. Para ello seleccionaremos el botón y le pondremos como color de fondo el negro. Para ello nos valdremos de la pestaña de color.

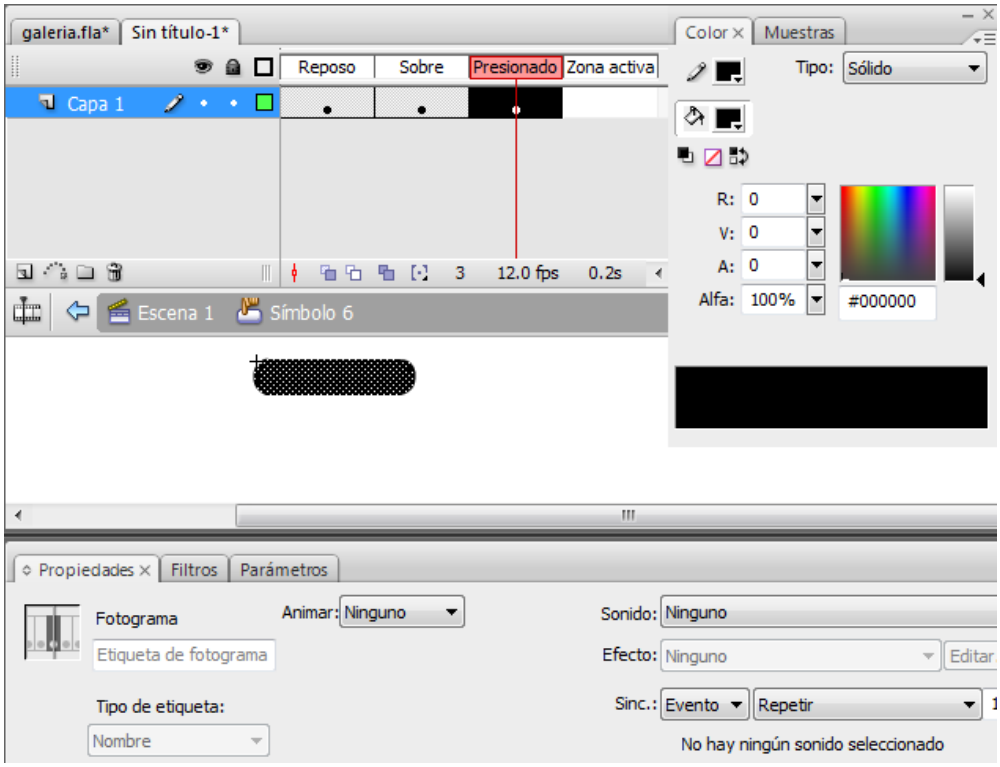


Figura 3.2.5. Galería Flash: Crear botón, Presionado

Para editar el texto que contendrá el botón, lo primero será realizar una copia en la biblioteca del botón que acabamos de crear, para que de este modo podamos coger tantos botones con los mismos estilos como queramos.

Volveremos a la edición del nuevo botón que hemos creado, pero esta vez crearemos una nueva capa en la pestaña de capas. En esta nueva capa crearemos un fotograma clave por cada estado del botón. Una vez realizado este proceso, nos situaremos en el estado de Reposo y emplearemos la herramienta de texto del panel de herramientas para añadir el texto “Siguiente”, el cual lo colocaremos en la posición XY (20,2) para que este se mantenga centrado en el centro del botón y haremos que esté en negrita para que resalte un poco el texto.

Emplearemos el mismo proceso en el estado Sobre y Presionado, salvo que en el estado Presionado el color del texto en vez de negro será blanco, para obtener la misma sensación que si se pulsa el botón “Siguiente” en Director.

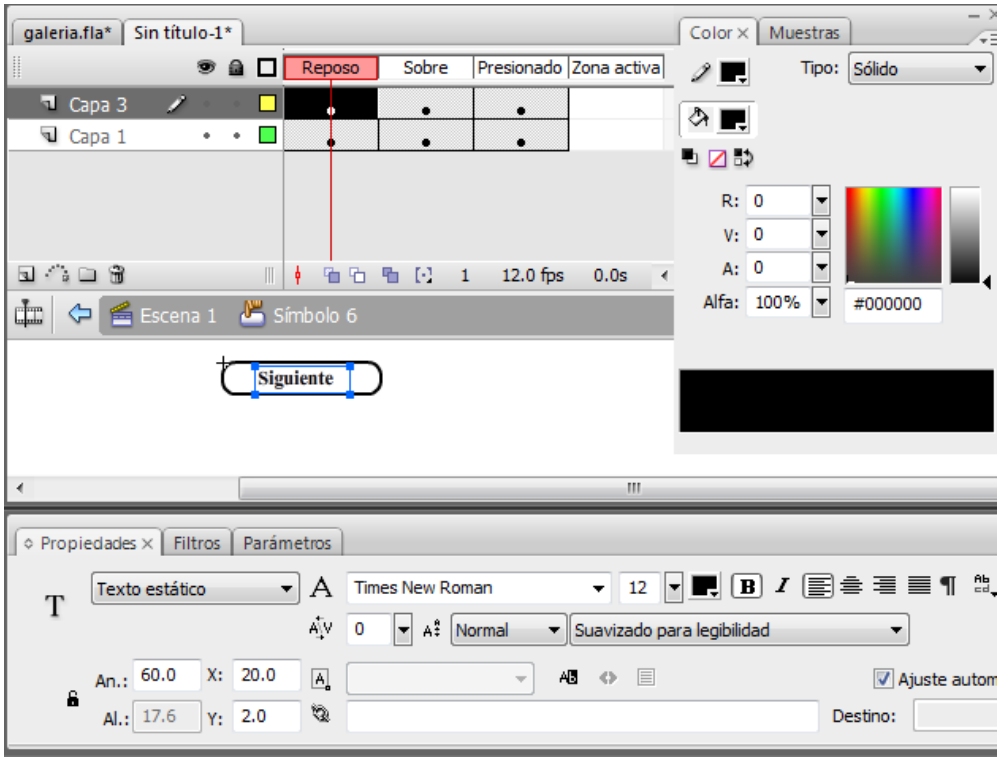


Figura 3.2.6. Galería Flash: Crear botón, Añadir Texto al botón

Repetiremos este proceso tantas veces como sea necesaria hasta añadir todos los botones que necesitemos para crear la galería.

Del mismo modo que se han añadido las imágenes, se añadirá la película que se quiera reproducir.

-
- **Creando las capas de trabajo:**

En el caso de flash, el conjunto del total de capas (pequeñas películas independientes) representarán la galería. En nuestro caso emplearemos un total de dos capas, una para representar los botones y otra para representar las imágenes y la película.

En cada capa emplearemos 12 fotogramas para simular la galería. El primer fotograma se empleará como presentación, los fotogramas 2-10 se emplearán para la galería de fotos y los fotogramas 11-12 para el reproductor.

Una vez creada y seleccionada la capa de botones, pulsaremos el botón F9 para acceder a la ventana de acciones, en la cual asociaremos a la capa seleccionada la siguiente función para que cada vez que entremos en un nuevo fotograma, la película se detenga:

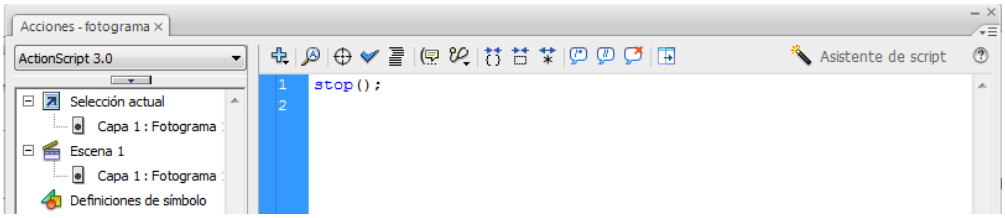


Figura 3.2.7. Galería Flash: Acción de Capa 1

- **Contenido del fotograma 1: Presentación**

El primer paso será crear un fotograma clave al comienzo de la línea del tiempo de la capa que se ha creado, es decir, en el fotograma 1.

Una vez seleccionado el fotograma, añadiremos las instancias de los elementos de texto, galería y reproductor que contiene la biblioteca, obteniendo una apariencia parecida a la que se creo en Director.

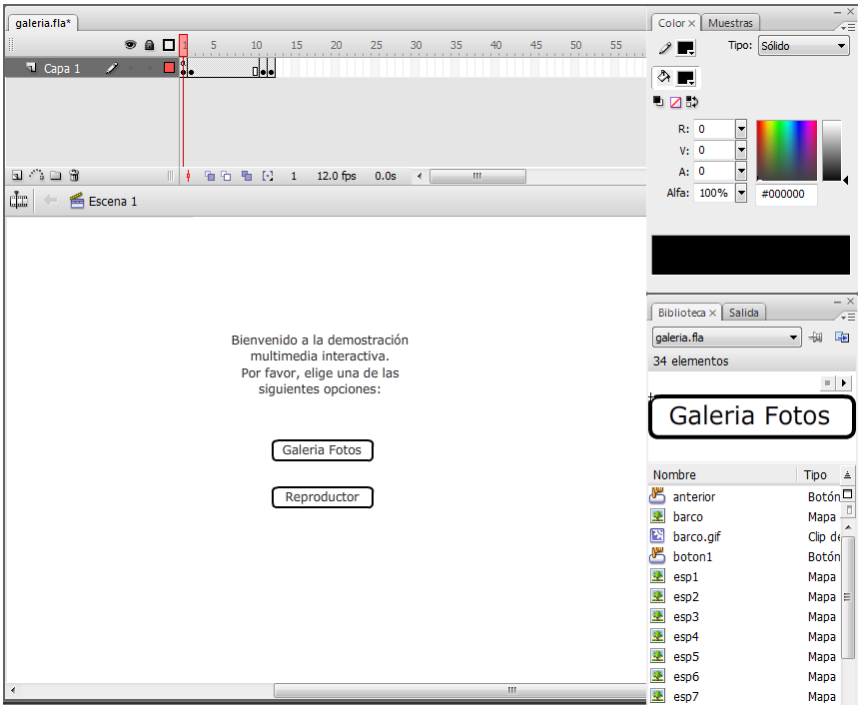


Figura 3.2.8. Galería Flash: Contenido fotograma 1

Una vez obtenida la apariencia deseada, con la herramienta de selección de la ventana de herramientas, seleccionar los botones para asignarles un nombre a cada instancia.

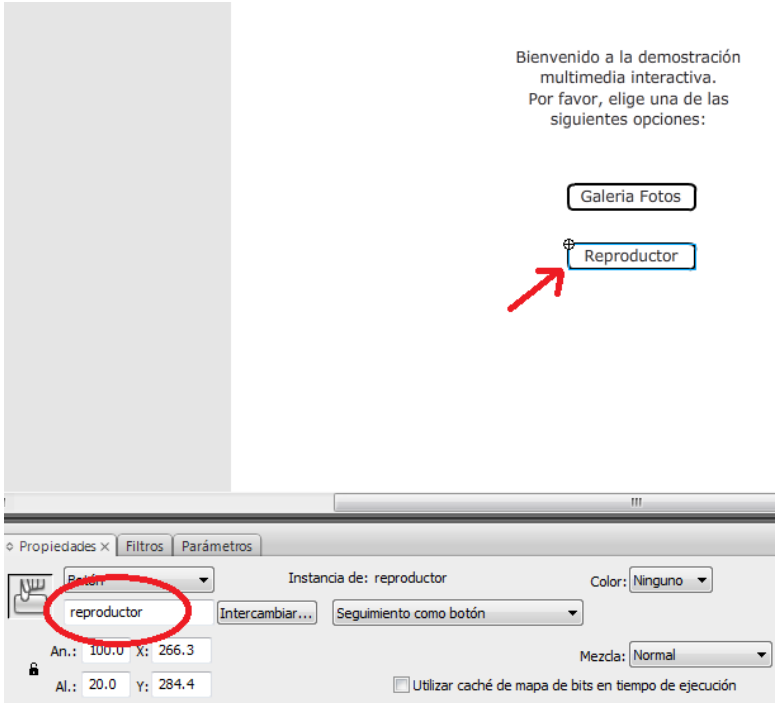


Figura 3.2.9. Galería Flash: Nombres de Instancias

Ahora que las instancias de estos botones pueden ser invocadas, añadiremos al código anterior eventos para que cuando se pulse un botón nos dirija al fotograma que corresponda cada uno. En el caso del botón de "Galería" nos dirigirá al fotograma 2 y en el caso del botón "Reproductor" nos dirigirá al fotograma 11.

```
1  stop();
2  galeria.addEventListener(MouseEvent.CLICK, onClickGoToGaleria);
3  reproductor.addEventListener(MouseEvent.CLICK, onClickGoToReproductor);
4
5  function onClickGoToGaleria(event:MouseEvent):void {
6      gotoAndStop(2);
7  }
8
9  function onClickGoToReproductor(event:MouseEvent):void {
10     gotoAndStop(11);
11 }
```

- **Contenido del fotograma 2-10: Galería de Fotos**

En estos fotogramas ilustraremos un menú de navegación y se tendrán que visualizar las fotos que están importadas en la biblioteca en función de la posición de la imagen central en el navegador. También tendremos la opción de partir al reproductor.

En este caso navegaremos por los fotogramas y en función del fotograma sobre el que estemos localizados se visualizarán unas imágenes u otras. En el primer fotograma (el segundo fotograma de la película en este caso), visualizaremos las imágenes que se mostrarán inicialmente. Cada fotograma sucesor representará la siguiente imagen a mostrar. De este modo continuaremos hasta que se muestre la imagen final.

El primer paso consistirá en crear un fotograma clave sobre el segundo fotograma de la película en la capa de imágenes y películas. A continuación, se añadirá en el centro

de la escena la imagen que se quiera visualizar como principal en la galería y añadir otra vez la misma imagen en miniatura para la navegación. Al lado de la imagen de navegación, añadiremos la siguiente imagen que se mostrará centrada en el siguiente fotograma, pero en este caso en miniatura. Por último, en la capa de botones se añadirán los botones de "Siguiente" y "Anterior" para la navegación de la galería y el botón de "Reproductor" para poder acceder al reproductor en cualquier momento.

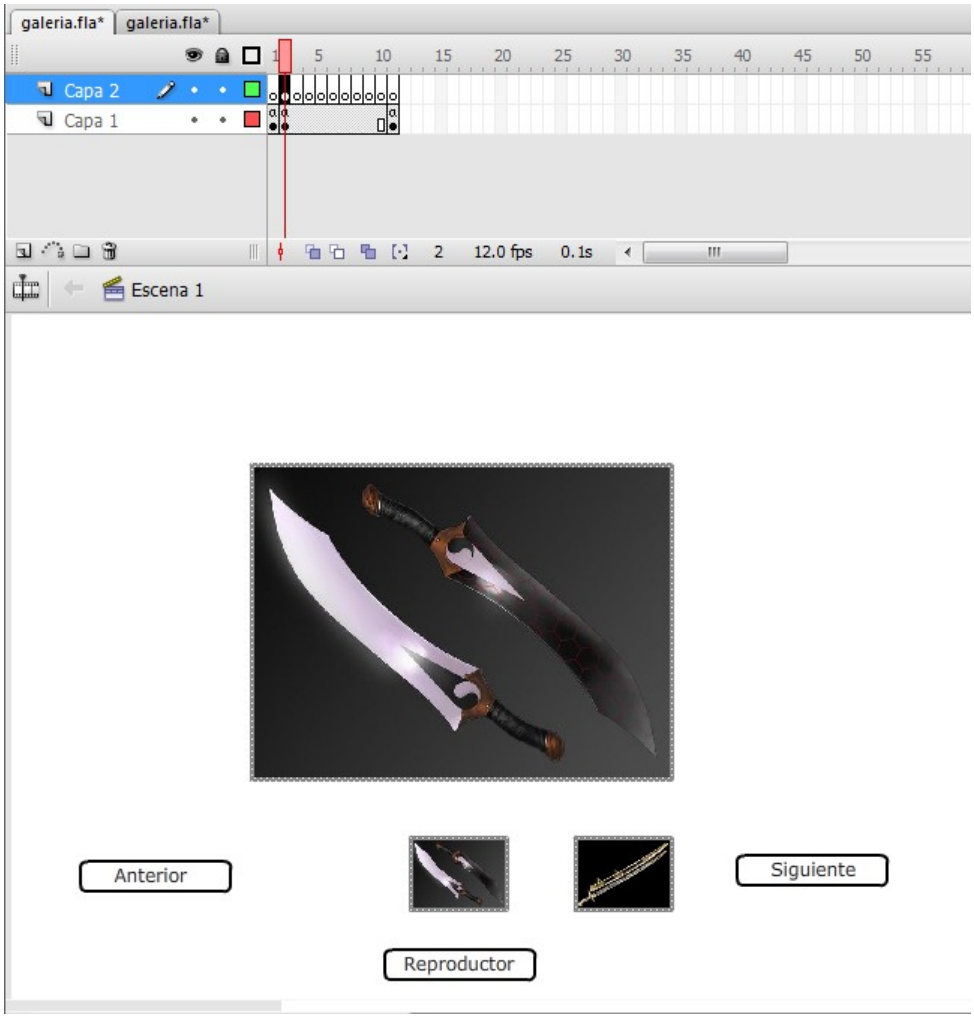


Figura 3.2.10. Galería Flash: Contenido fotogramas 2-10

Una vez terminada la interfaz que queramos ofrecer, continuaremos del mismo modo con el siguiente fotograma, de tal modo que debe obtener la apariencia que deseemos que logre al pulsar el botón "Siguiente".

Deberemos continuar con este proceso hasta que hayamos completado todos los fotogramas correspondientes a la galería.

Por último, se tienen que añadir a la capa de botones las acciones que nos permitirán navegar por la galería.

Para el botón "Anterior" retrocederemos un fotograma de la película, y de este modo, retrocederemos en la película un nivel y se mostrará la imagen deseada.

Para el botón “Siguiente” avanzaremos un fotograma de la película y se visualizará la siguiente imagen de la galería.

Por último, para el botón “Reproductor”, nos enviará al fotograma en el cual comienza el reproductor de la galería.

```
stop();
var a:Number = 2;
anterior.addEventListener(MouseEvent.CLICK, onClickAnterior);
siguiente.addEventListener(MouseEvent.CLICK, onClickSiguiente);
reproductor2.addEventListener(MouseEvent.CLICK, onClickGoToReproductor2);

function onClickSiguiente(event:MouseEvent):void {
    a=a+1;
    if(a>10){
        a=10;
    }
    gotoAndStop(a);
}

function onClickAnterior(event:MouseEvent):void {
    a=a-1;
    if(a<2){
        a=2;
    }
    gotoAndStop(a);
}

function onClickGoToReproductor2(event:MouseEvent):void {
    gotoAndStop(11);
}
```

- **Contenido de los fotogramas 11-12: Reproductor**

En estos últimos fotogramas, visualizaremos una pequeña película, un GIF en este caso, pero podríamos hacer lo mismo con una película distinta.

Para la capa de imágenes y vídeo, introduciremos un nuevo fotograma clave, en el cual añadiremos el clip de película que tenemos importado en la biblioteca, mientras que en la capa de botones añadiremos los botones de “Play”, “Pause”, “Stop” y “Reproductor”.

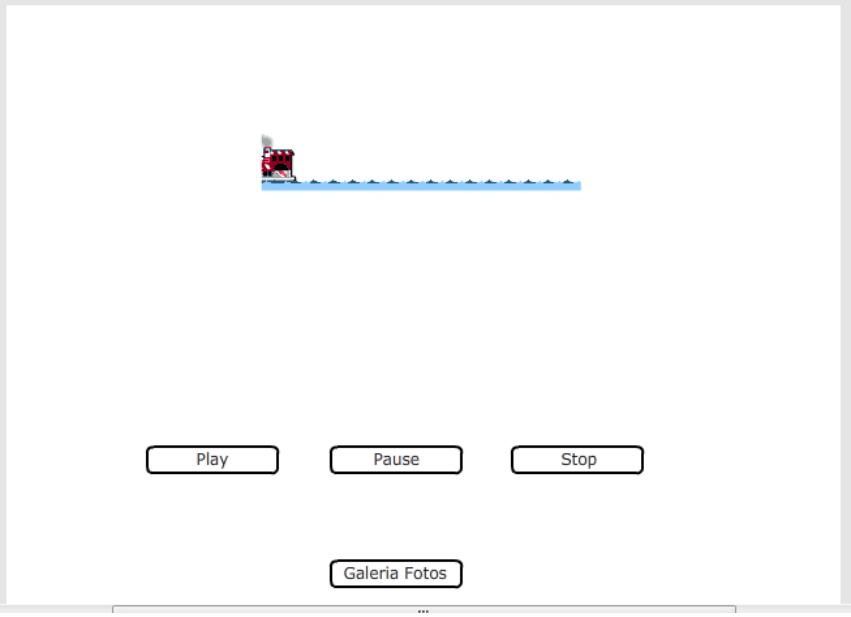


Figura 3.2.11. Galería Flash: Contenido fotogramas 11-12

El último paso será añadir los eventos correspondientes a los botones.

Para "Galería", nos devolverá al fotograma en el que comienza la galería de fotos.

Para "Play", accederemos a la instancia de la película y emplearemos el evento play() que nos proporciona Action Script.

Para "Pause", emplearemos el evento stop() de la instancia del clip de película que nos proporciona otra vez Action Script.

Por último, para "Stop" avanzaremos de fotograma, para después volver al mismo, de tal modo que la ventana pueda refrescarse correctamente.

```
1  stop();
2  barco.stop();
3  galeria2.addEventListener(MouseEvent.CLICK, onClickGoToGaleria2);
4  botonPlay.addEventListener(MouseEvent.CLICK, onClickPlay);
5  botonPause.addEventListener(MouseEvent.CLICK, onClickPause);
6  botonStop.addEventListener(MouseEvent.CLICK, onClickStop);
7
8  function onClickGoToGaleria2(event:MouseEvent):void {
9      gotoAndStop(2);
10 }
11
12 function onClickPlay(event:MouseEvent):void {
13     barco.play();
14 }
15
16 function onClickPause(event:MouseEvent):void {
17     barco.stop();
18 }
19
20 function onClickStop(event:MouseEvent):void {
21     gotoAndPlay(12);
22 }
```

- **Generando la película**

Para obtener el ejecutable con nuestra galería de fotos lo único de deberemos hacer es acceder a la pestaña Archivo →Publicar. Automáticamente se un HTML (Ya que flash está pensado para Web), con todas las librerías que este necesite.

3.3 Observaciones

- Flash permite un desarrollo de los elementos más elaborado frente a Director.
- Flash no requiere incluir un objeto necesariamente en la biblioteca para que interactúe en la película.
- Ambos son capaces de asociar código a un nivel profundo y complejo.
- Las aplicaciones Flash resultantes, optimizan mejor el espacio que ocupan en memoria una vez publicadas. Este es un factor muy positivo si lo que queremos es crear una aplicación Web.
- La estructura de construcción de una película es idéntica en ambas herramientas.

Capítulo 4. MANEJO DE BASES DE DATOS SOBRE LAS HERRAMIENTAS AUTOR:

En este capítulo se analizará el modo en el que estas dos herramientas interactúan con una Base de Datos, tanto de manera local como remota. Se creará una aplicación de consola de comandos SQL para su ilustración.

4.1 Macromedia Director MX 2004

4.1.1 Empleo de BBDD en local

- **Xtra XMySQL**

Macromedia Director MX 2004 no dispone de ninguna herramienta integrada para facilitarnos el acceso a bases de datos, pero en cambio permitió que aquellos programadores que quisieran añadir funcionalidades al programa, pudieran hacerlo mediante la creación de Xtras.

Un Xtra es un plug-in modular, que permite añadir nuevas capacidades o funcionalidades a Macromedia. Para que un Xtra pueda ser instalado, lo único que se debe hacer es colocarlo en la carpeta Xtra de la aplicación de Director.

En nuestro caso, para acceder a una base de datos de manera local, disponemos de varios xtras a nuestra disposición, los cuales se pueden obtener de la página oficial de Xtrasy: http://xtrasy.com/index.cfm?module=xtras&action=listxtras&cat=13

El xtra que emplearemos para acceder a una base de datos local será **xmysql**. Este xtra accederá directamente a los ficheros binarios de un MySQL y realizará consultas sobre este sin la necesidad de un ODBC.

Cuando descarguemos el xtra, veremos que este, además del fichero xmysql.xtra, contiene un MySQL propio y un pdf con la documentación del xtra.

Las funciones que emplearemos del api del Xtra serán:

-Función para comprobar que el proceso MySQL está lanzado:

```
Integer running = Xm_ServerRunning(String fileName)
```

Donde fileName es el servicio MySQL. Si el servicio está lanzado, devolverá 1, en caso contrario, 0.

-Función para lanzar el servidor MySQL local:

```
Integer success = Xm_ServerStart(String commandLine)
```

Donde commandLine será el comando que lance el servidor MySQL. Si la ejecución del comando ha sido satisfactoria, devolverá 1, en caso contrario, devolverá 0.

-Función para cargar el xtra xmysql para realizar conexión a base de datos

```
Set connObject = new (xtra "xmysql")
```

Donde connObject será el objeto que contenga los datos del xtra xmysql

-Función para realizar la conexión a la base de datos local:

```
Integer success = ConnObject.Xm_Connect(String host, String
user, String pass, String DefaultDB, Integer port);
```

Donde ConnObject es el objeto que contiene los datos cargados del xtra xmysql, host es la localización del servidor MySQL, user es el usuario de acceso, pass es la contraseña de acceso, defaultDB es el nombre del schema al que nos conectaremos y port el puerto del servidor MySQL. En caso de realizar una conexión satisfactoria, devolverá 1, en caso contrario 0.

-Función para manejarnos entre los distintos schemas de la base de datos:

```
ConnObject.Xm_SelectDB(String schema)
```

Donde ConnObject es el objeto que contiene los datos cargados del xtra xmysql y schema es el schema al que queremos conectarnos.

-Función para detener el servidor MySQL:

```
Integer success = Xm_ServerKill(String fileName)
```

Donde fileName es el servicio MySQL. Si se ha detenido el servicio de manera satisfactoria, devolverá 1, en caso contrario, 0.

-Función para ejecutar SLQ:

```
Integer resultSerId = ConnObject.Xm_Query(String SQL)
```

Donde ConnObject es el objeto que contiene los datos cargados del xtra xmysql y SQL es la query SQL que ejecutaremos. En caso de que la función falle, devolverá 0, de lo contrario, devolverá un número con el registro del resultado que podremos emplear más adelante.

-Función para recoger los mensajes de error:

```
List ErrorMessage = ConnObject.Xm_GetError()
```

Donde ConnObject es el objeto que contiene los datos cargados del xtra xmysql. Devolverá la traza de error que se ha capturado.

-Función de procesado de respuesta de la consulta para obtener la cantidad de resultados:

```
Integer rowCount = ConnObject.Xm_RowCount();
```

Donde ConnObject es el objeto que contiene los datos cargados del xtra xmysql. Devolverá la cantidad de filas que ha generado la consulta. En caso de error, devolverá -1.

-Función de procesado de respuesta de la consulta para obtener la lista de los nombres de los campos:

```
List fieldList = ConnObject.Xm_FetchFieldList(integer result-
SetId)
```

Donde ConnObject es el objeto que contiene los datos cargados del xtra xmysql y resultSetId es el registro del resultado de la consulta SQL. Devolverá una lista con los nombres de los campos del resultado de la consulta. En caso de error la lista estará vacía.

-Función de procesado de respuesta de la consulta para obtener la lista de los resultados de los campos:

```
List fieldValueList = ConnObject.Xm_FetchRowList(integer res-
ultSetId)
```

Donde ConnObject es el objeto que contiene los datos cargados del xtra xmysql y resultSetId es el registro del resultado de la consulta SQL. Devolverá una lista con los valores de los campos del resultado de la consulta. El último contenido de la lista será [“XM_EOF”]. En caso de error la lista contendrá un único valor: [“XM_ERROR”].

-Función de procesado de respuesta de la consulta para liberar resultados:

```
Integer success = ConnObject.Xm_FreeResult(integer resultSet-
Id)
```

Donde ConnObject es el objeto que contiene los datos cargados del xtra xmysql y resultSetId es el registro del resultado de la consulta SQL. Si logró liberar la memoria devolverá 1, en caso contrario 0.

- **Aplicación Consola de Comandos MySQL**

(**NOTA**: Antes de ejecutar la aplicación Macromedia Director MX 2004, añadir xmysql.xtra a la carpeta de xtras de Macromedia Director MX 2004.)

Para la creación de esta aplicación emplearemos dos fotogramas: Uno para la consola de comandos MySQL y para indicar que no se pudo realizar la conexión a la base de datos.

En el primer fotograma emplearemos un elemento de texto para representar los resultados de los comandos SQL que empleemos, un field para escribir los comandos SQL y un botón para ejecutar los comandos SQL.

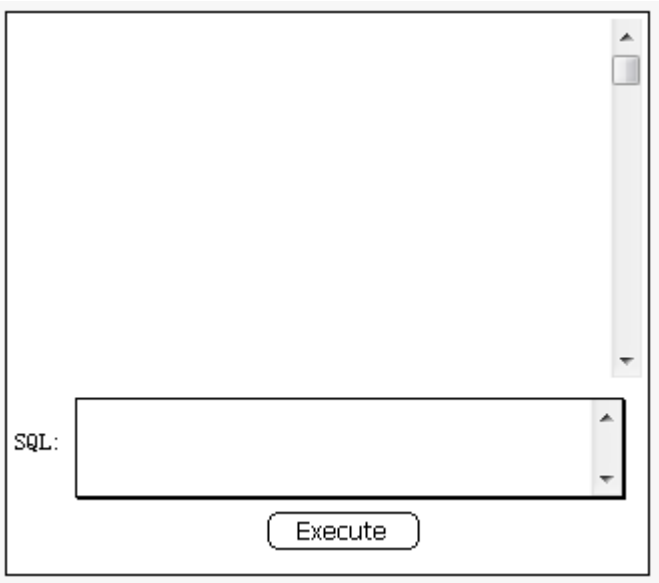


Figura 4.1.1.1. Director: BD local, fotograma 1

En el segundo fotograma, emplearemos un elemento de texto para representar el error de conexión a la base de datos.

Figura 4.1.1.2. Director: BD local, fotograma 2

Como siempre, crearemos dos scripts a nivel de fotograma para que la película se detenga al comienzo de cada fotograma para poder navegar por este.

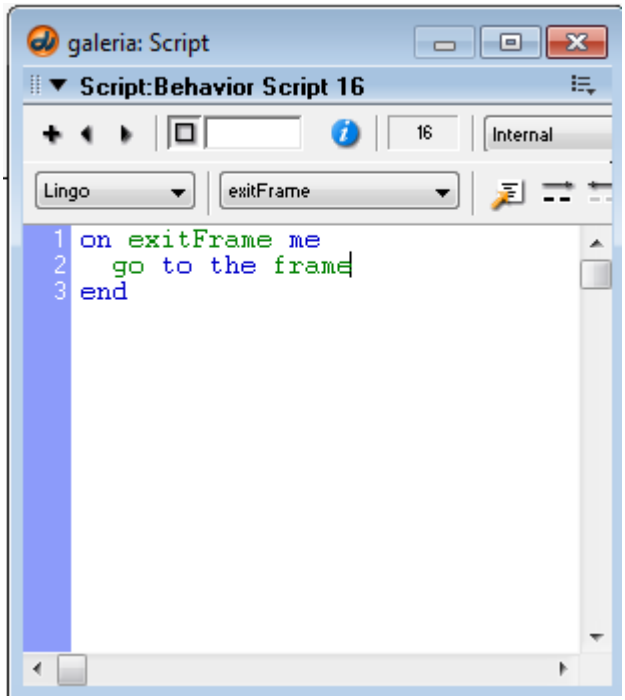


Figura 4.1.1.3. Director: BD local, Script a nivel de Fotograma

Ahora que disponemos del esqueleto de la aplicación, el siguiente paso será hacer que al comienzo de la película, realice la conexión a la base de datos. Para ello emplearemos un script a nivel de Movie.

En el script se comprobará si el servicio MySQL está lanzado, y en caso de que el servicio no lo este, la aplicación lo lanzará. Una vez realizada la comprobación, se encargará de establecer la conexión con la base de datos. En caso de que alguno de estos dos pasos produzca algún error, la película se situará en el fotograma de la pantalla de error y advertirá cuál ha sido el problema. Al finalizar la película, se deberá terminar con el proceso del servidor MySQL.

```
--Variables globales que representaran mysql="la conexion a
la BD" y res="existe un resultado (1 o 0)"
global mysql,res
--Cuando comience la ejecución del programa lanzamos la BD y
nos conectamos
on startMovie
  global mysql
  --chequea si el servidor mysql esta lanzado, sino lo
lanzamos
  if Xm_ServerRunning("mysqld-opt.exe")=0 then
    if Xm_ServerStart(quote&(the
moviepath)&"mysql\bin\mysqld-opt.exe"&quote&" --no-defaults
--basedir="&quote&(the moviepath)&"mysql"&quote&"
--port=10111")=0 then
      --Si no podemos lanzarlo monstraremos un mensage de
texto indicando que no hemos podido iniciar el servidor y
terminaremos la ejecucion del programa
      set the text of member "ErrorText" to "No se puede
ejecutar el servidor, asegurese de que la carpeta mysql y el
extra xmysql estén en el mismo directorio que la aplicación"
```

```
      go to "Error"
      exit
    end if
  end if
  --Ahora realizaremos la conexión la la base de datos
cargando el extra de xmysql
  set mysql= new (xtra "xmysql")
  --Nos conectamos a la base de datos local
  if mysql.Xm_Connect("localhost","root","root","",10111)=0
then
    --En caso de no poder conectarse mostrará un texto
indicando que no a podido realizarse la conexión y el
programa se cerrará
    set the text of member "ErrorText" to "Can't connect to
the server"
    go to "Error"
    exit
  end if
  --Seleccionamos la BD si no lo hemos hecho en la Xm_Connect
  mysql.Xm_SelectDB("xmysql")
  --Dejamos en blanco la casilla de texto queryresult
  set the text of member "queryresult" to ""
end startMovie
--Cuando terminemos la ejecución de la pelicula matamos el
proceso mysql-opt.exe y la referencia al xtra xmysql
on stopMovie
  global mysql
  set mysql = 0
  Xm_ServerKill("mysqld-opt.exe")
end stopMovie
```

Por último, crearemos un script a nivel de member para el botón que ejecuta las SQL. Este script recogerá el contenido del field que contiene la SQL y ejecutará la query. Se encargará de visualizar el resultado de la query en el elemento de texto para resultados, ya sean resultados válidos, o errores de consulta.

```
on mouseUp me
  global mysql
  -- Ejecutamos la consulta que hay en el text member "sql"
  res = mysql.Xm_Query(the text of member "sql")
  -- Si el resultado es 0, esque o bien a ocurrido un error o
bien la consulta no era un SELECT (es decir, seria un create,
update...)
  if res=0 then
    -- Almacenamos el error
    ErrMsg=mysql.Xm_GetError()
    -- Si resulta que no habia devuelto valores, mostraremos
que no hay valores en el text member "queryresult"
    if ErrMsg[1]="XmySQL-->No result set returned." then
      set the text of member "queryresult" to "No hay valores
de retorno. Puede que no haya introducido una sentencia
SELECT."
```

```
      -- Si resulta que era un error lo mostrará en el text
member "queryresult"
    else
      set the text of member "queryresult" to ErrMsg[2]
    end if
    -- Si el resultado es distinto de 0 lo mostraremos
  else
    -- Primero limpia el text member "query result"
    set the text of member "queryresult" to ""
    -- Empleamos una variable para almacenar temporalmente la
respuesta
    tmp = ""
    --Obtener el numero de registros(la cantidad de filas)
    rows = mysql.Xm_RowCount(res)
    if rows>0 then
      -- Obtener la lista con los nombres de los campos
      FieldList = mysql.Xm_FetchFieldList(res)
      -- Obtener cuentos campos tiene la consulta
      fields = mysql.Xm_FieldCount(res)
      -- Mostrar el nombre de los campos
      repeat with i = 1 to fields
        -- Hacer que text member "queryresult" =
"queryresult"&FieldList[i]&tab (es decir, ir añadiendole la
cabecera a la consulta)
        tmp = tmp & FieldList[i]&tab
      end repeat
      --Dejar el text member "queryresult" =
"queryresult"&"salto de linea"
      tmp = tmp & return
      --Obtener el contenido de la consulta fila por fila
      repeat with k = 1 to rows
        -- Obtener la lista con los resultados de esa linea
        row=mysql.Xm_FetchRowList(res)
        repeat with i = 1 to fields
          -- Lo añadimos al text member "queryresult"
          tmp = tmp&row[i]&tab
        end repeat
        -- Volvemos a añadirle otro salto de linea
        tmp=tmp&return
      end repeat
      -- Mostrar el resultado
      set the text of member "queryresult" to tmp
    else
      -- Si no habia resultados para mostrar...
      set the text of member "queryresult" to "Ningún res-
ultado que mostrar"
    end if
    -- Liberamos la variable de resultado
    mysql.Xm_FreeResult(res)
  end if
end
```

- **Modo de empleo de la aplicación:**

Una vez generada la película, es importante recordar que el ejecutable siempre debe de ir acompañado de la carpeta MySQL, ya que hemos definido la ruta del servidor de MySQL en función de la localización de la película.

Una vez lanzada la película, podemos crear una nueva tabla:

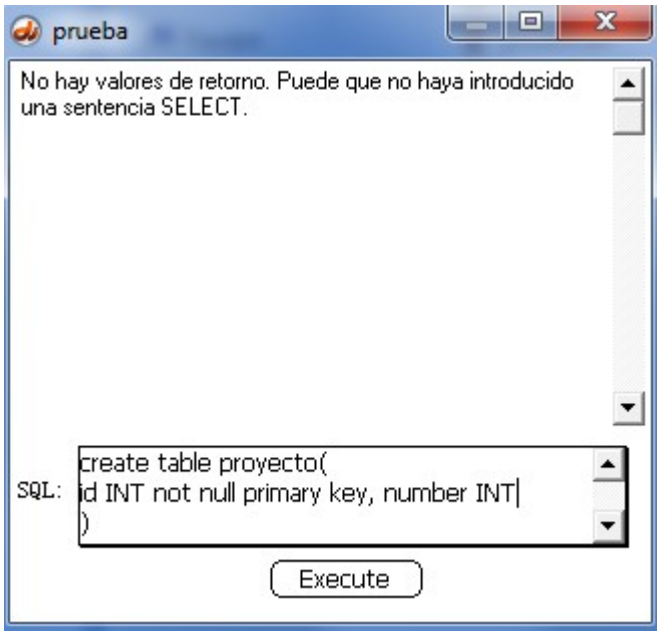


Figura 4.1.1.4. Director: BD local, Modo de empleo 1

Realizar Inserciones:

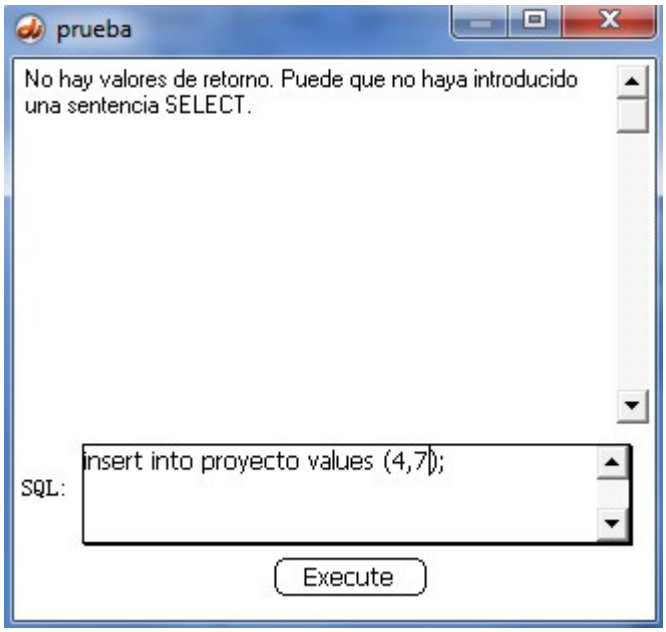


Figura 4.1.1.5. Director: BD local, Modo de empleo 2

O ejecutar consultas:

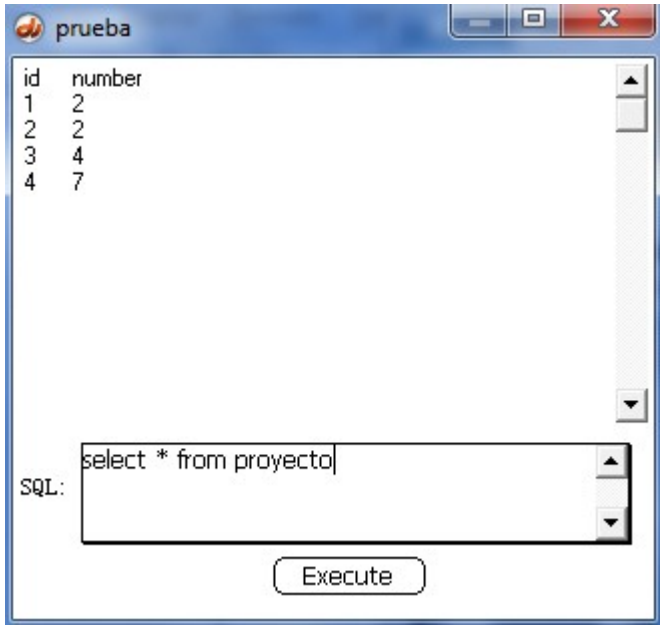


Figura 4.1.1.6. Director: BD local, Modo de empleo 3

Y si no encuentra el directorio MySQL:

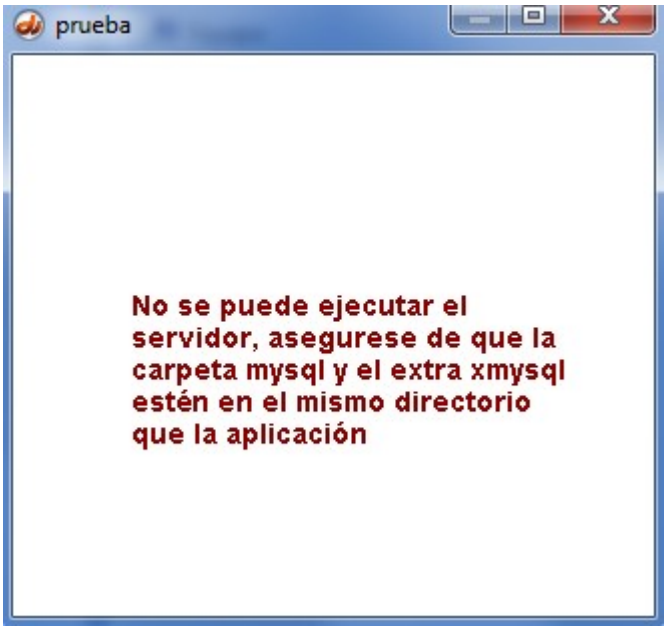


Figura 4.1.1.7. Director: BD local, Modo de empleo 4

4.1.2 Empleo de BBDD en Servidor

- **Xtra XMySQL**

Macromedia Director MX 2004 posee muchos Xtras, pero ninguno todavía que sea capaz de acceder a una base de datos remota. De todos modos, el Xtra xmysql, nos permite acceder al MySQL que nos proporciona, de manera remota, si accedemos a través de la misma red de área local.

- **Aplicación Consola de Comandos MySQL**

Partiremos de la misma aplicación del apartado anterior para crear la consola de comandos. La única modificación que aplicaremos será en el Movie Script, el cual ya no se encargará de comprobar si el servidor MySQL está lanzado. Simplemente intentará realizar una conexión al destino especificado y en caso de no conseguirlo, nos devolverá al fotograma de "Error":

```
global mysql,res
on startMovie
  global mysql
  set mysql= new (xtra "xmysql")
  if mysql.Xm_Con-
nect("192.168.1.45","root","root","",10111)=0 then
    set the text of member "ErrorText" to "Can't connect to
the server"
    go to "Error"
    exit
  end if
  mysql.Xm_SelectDB("syntax")
  set the text of member "queryresult" to ""
end startMovie
```

El resto de la película será exactamente la misma, por lo que la aplicación ya estará lista para ser ejecutada en cualquier ordenador en la misma red de área local en la que se encuentre el servicio MySQL.

4.2 Adobe Flash CS3

4.2.1 Empleo de BBDD

Flash fue creado para crear aplicaciones que corrieran en la Web, por lo que para acceder a una base de datos, requiere interactuar con PHP y este requiere de un servidor, por lo que crearemos un entorno de desarrollo sobre un servidor Apache.

- **Entorno WAMP**

El primer paso será instalar un WAMP (Windows + Apache + MySQL + PHP) que podemos obtener del sitio: http://www.appservnetwork.com/. Una vez descargado el instalador, lo ejecutaremos e instalaremos todos los componentes que necesitemos:

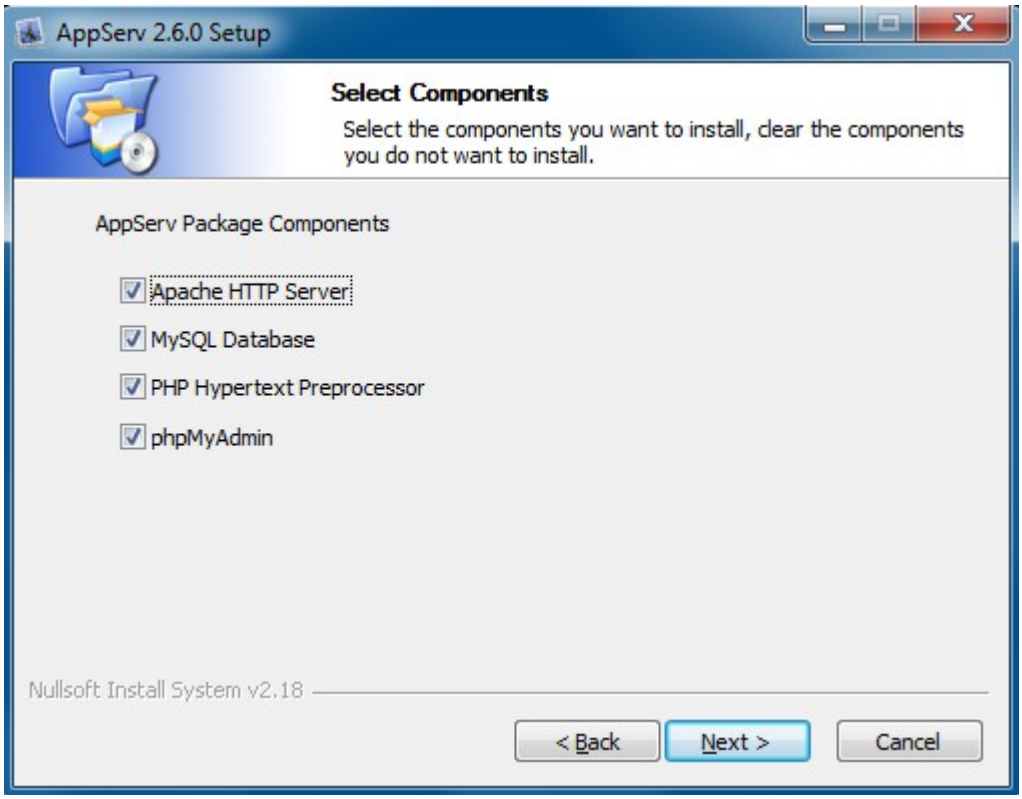


Figura 4.2.1.1. Entorno WAMP: Instalación

El siguiente paso importante será especificar la contraseña del usuario root, el usuario que manejará todos los permisos de la base de datos.

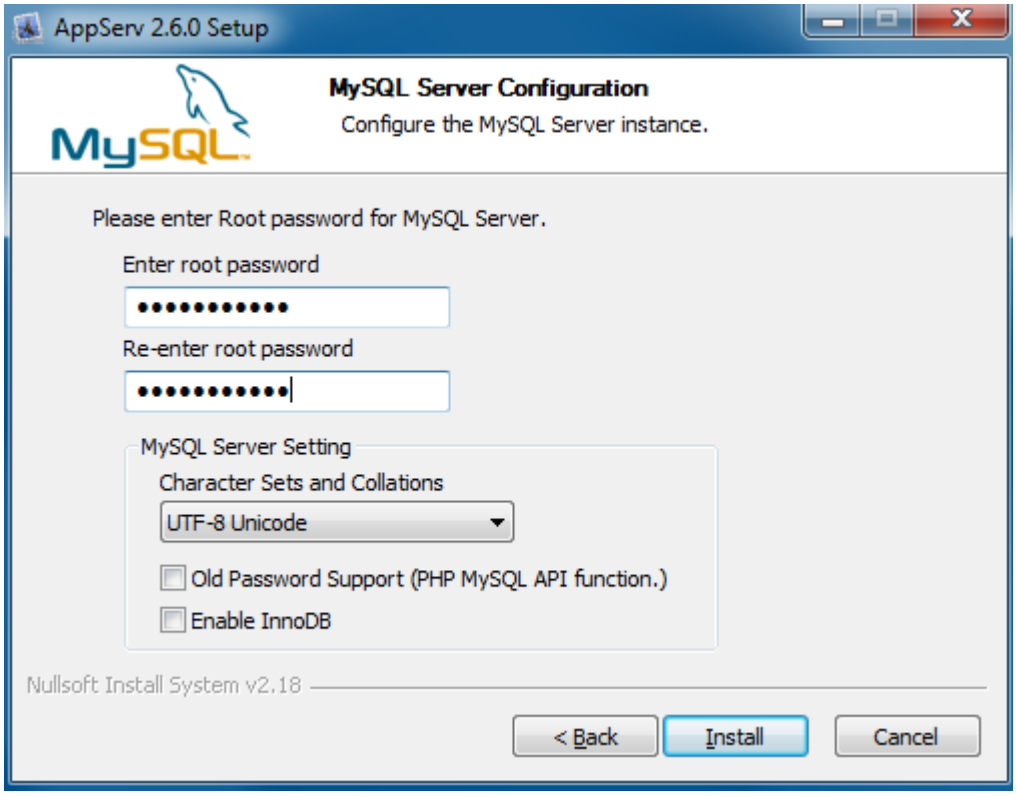


Figura 4.2.1.2. Entorno WAMP: configuración

Una vez realizada la instalación, accederemos a la URL: http://127.0.0.1/phpMyAdmin/ para comprobar que se ha instalado correctamente:

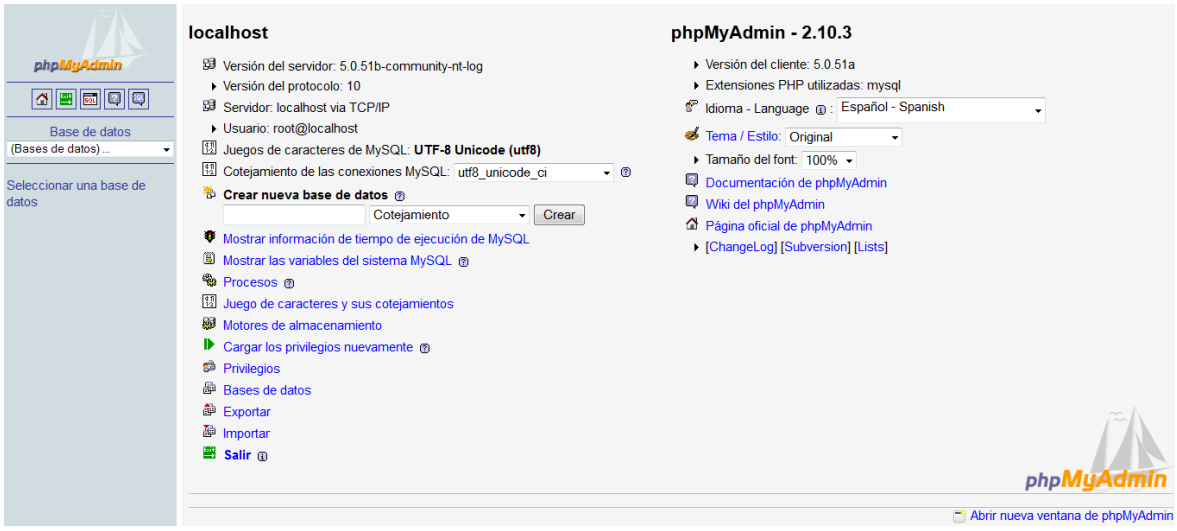


Figura 4.2.1.3. Entorno WAMP: phpMyAdmin

A partir de este punto, nuestra área de trabajo se localizará en DIR_APPSERV/www/DIRECTORIO_TRABAJO.

- **Aplicación Consola de Comandos MySQL**

Al contrario que en la aplicación en Director, en este caso crearemos una aplicación que empleará un único fotograma y en una sola capa. En el fotograma se visualizará la consola de comandos SLQ.

Para visualizar el fotograma correctamente, crearemos un botón para ejecutar las consultas SQL, un elemento de texto multilínea en el que se escribirán los comandos SQL y un elemento de texto dinámico sobre el que mostraremos los resultados de las consultas SQL:

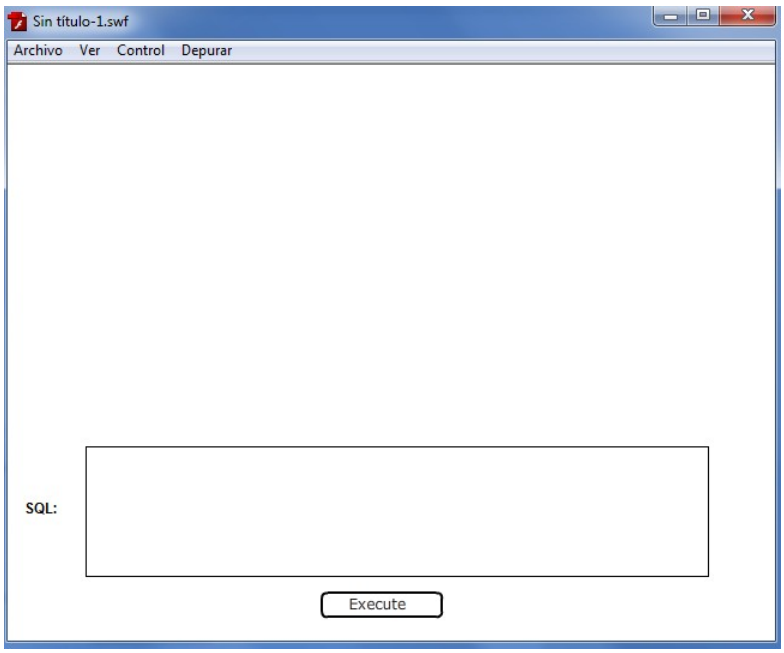


Figura 4.2.1.4. Flash, BD local: Fotograma clave

Ahora que disponemos del esqueleto de la aplicación, el siguiente paso será enviar al servidor la petición con la consulta SQL. Para esta funcionalidad, Flash dispone de la clase LoadVars, que sirve para la transferencia de variables entre Flash y un servidor Web a través de HTTP.

Con la clase LoadVars podemos enviar todas las variables de un objeto a una URL y cargar todas las variables de una URL en un objeto. También permite enviar variables concretas en lugar de todas las variables, lo que puede contribuir a que la aplicación sea más eficaz.

Emplearemos la clase LoadVars para que interactúe entre Flash y un servicio de PHP, el cual se encargará de realizar las consultas sobre el servidor MySQL.

De la clase LoadVars emplearemos las siguientes funciones:

-Función para crear una instancia del objecto LoadVars:

```
Var loadVarObject:LoadVars = new LoadVars();
```

En loadVarObject tendremos la instancia del objeto LoadVars.

-Función para enviar el contenido de un objeto LoadVars a una URL:

```
Boolean success = sendLoadVarsObject.sendAndLoad(String url,
Object returnLoadVarsObject, [String method])
```

Donde sendLoadVarsObject es una instancia de LoadVars que contiene los elementos a enviar, url es el destino de la petición http que se realizará, returnLoadVarsObject es la instancia de LoadVars en el que se procesará el resultado y method es el método de envio que se empleará (GET o POST). Si el proceso fue satisfactorio, devolvera TRUE, en caso contrario, FALSE.

-Función que se podrá invocar cuando se ha finalizado la operación sendAndLoad() para tratar el resultado de la petición HTTP:

```
returnLoadVarsObject.onLoad = function(Boolean success){}
```

Donde returnLoadVarsObject es la instancia del objecto LoadVars sobre el que se procesará el resultado de la petición, tanto si ha sido satisfacttoria, en caso de contener como valor de retorno TRUE, o no, en caso de FALSE.

Una vez conocidas las funciones que emplearemos, definiremos los métodos que capturen la pulsación del botón, para que luego se realice la petición http con la SQL y mostrar los resultados.

Para el botón, llamaremos a la función que se encargará de realizar la petición y mostraremos un texto de “Cargando…” mientras se esperen los resultados

```
on (release) {
     // Colocamos un mensaje que indica que está cargando
     en // el campo de texto “mensaje_text”
     respuesta_txt.text = ”Cargando...”;
     //Llama a la función que envia el formulario
     enviarConsulta();
}
```

Sobre la capa, programaremos la función que envía la consulta y trata el resultado:

```
//Declaro las variables para enviar y para recibir
Var envio_lv:LoadVars = new LoadVars();
Var recibir_lv:LoadVars = new LoadVars();
//Función que enviará la consulta SQL
Function enviarConsulta() {
     // El nombre de la variable que enviara los datos del
     // formulario es "envio_lv", a esa variable le voy
     // asignando el valor de la query que le llegara al
     // php, y con el mismo nombre que asigno aqui
     // sera con el que los Scripts trataran los datos del
     // php
     envio_lv.consulta = consulta_txt.text;
     // Uso el metodo "sendAndLoad" para enviar la query
     // y recibir la respuesta del servidor, el metodo tiene
     // tres parámetros el primero es la URL del script que
```

```
        // tratara el formulario, en este caso lo llame
        // "form.php", el segundo es el objeto que cargara la
        // respuesta del servidor y el tercero el metodo de
        // envio del formulario que puede ser, como en HTML, GET
        // o POST
        envio_lv.sendAndLoad("http://localhost/cosulta/query.php
", recibir_lv, "POST");
}
//Función que procesa los datos recibidos del servidor
recibir_lv.onLoad = function(exito) {
        if (exito) {
                // Le asigna la respuesta recibida del servidor al
                // elemento de texto respuesta_txt
                if(this.mensaje == undefined
                   || this.mensaje == ""){
                     respuesta_txt.text = "La query no generó
        ningun resultado";
                } else {
                     respuesta_txt.text = this.mensaje;
                     this.mensaje = undefined;
                }
        } else {
                //Muestra un mensaje de error en negrita
                respuesta_txt.text = "Error en el php";
        }
};
```

En el siguiente paso debemos programar un php que recoja mediante POST la SQL que se le envía y procesarla, devolviendo una cadena que contenga el resultado que espera nuestro flash.

```
<?
        // variable donde almacenaremos la consulta
        $consulta=$_POST["consulta"];
        // el host de la base de datos
        $host = "localhost";
        // usuario de la base de datos
        $user = "root";
        // contraseña de la base de datos
        $pass = "root";
        // base de datos a usar
        $bbdd = "proyecto";
        // realizamos la conexión
        $conexion = mysql_connect($host,$user,$pass) or
die("mensaje=".mysql_error());
        // seleccionamos el schema
        mysql_select_db($bbdd,$conexion) or
die("mensaje=".mysql_error());
        // ejecutamos la query y obtenemos el resultado
        $result = mysql_query($consulta,$conexion) or
die("mensaje=".mysql_error());
        // variable donde almacenaremos la respuesta
```

```
	$respuesta = "";
	// variable que indicara el número del registro actual
	$numfield = 0;
	// realizaremos el recorrido de los resultados
	while($val=mysql_fetch_array($result)) {
		// mostramos los nombres de los campos
		if($numfield==0){
			$i=0;
			while($val[$i]!=null) {
				$campo = mysql_field_name($result, $i);
				if($i==0){
					$respuesta = $respuesta . "" .
$campo;
				} else {
					$respuesta = $respuesta . "\t" .
$campo;
				}
				$i++;
			}
			$respuesta = $respuesta . "\n";
		}
		$i=0;
		// mostramos el contenido de los registros
		while($val[$i]!=null) {
			if($i==0){
				$respuesta = $respuesta . "" . $val[$i];
			} else {
				$respuesta = $respuesta . "\t" .
$val[$i];
			}
			$i++;
		}
		$numfield++;
		$respuesta = $respuesta . "\n";
	}
	// si no ha habido resultados, informamos
	if($numfield==0){
	  $respuesta = $respuesta . "La query no ha generado
resultados.\n";
	}
	// devolvemos el resultado que contiene mensaje
	echo "mensaje=".$respuesta."";
?>
```

Ahora solo queda publicar la aplicación y ya estará operativa.

- **Modo de empleo de la aplicación:**

Una vez publicada la película, podemos crear una nueva tabla:

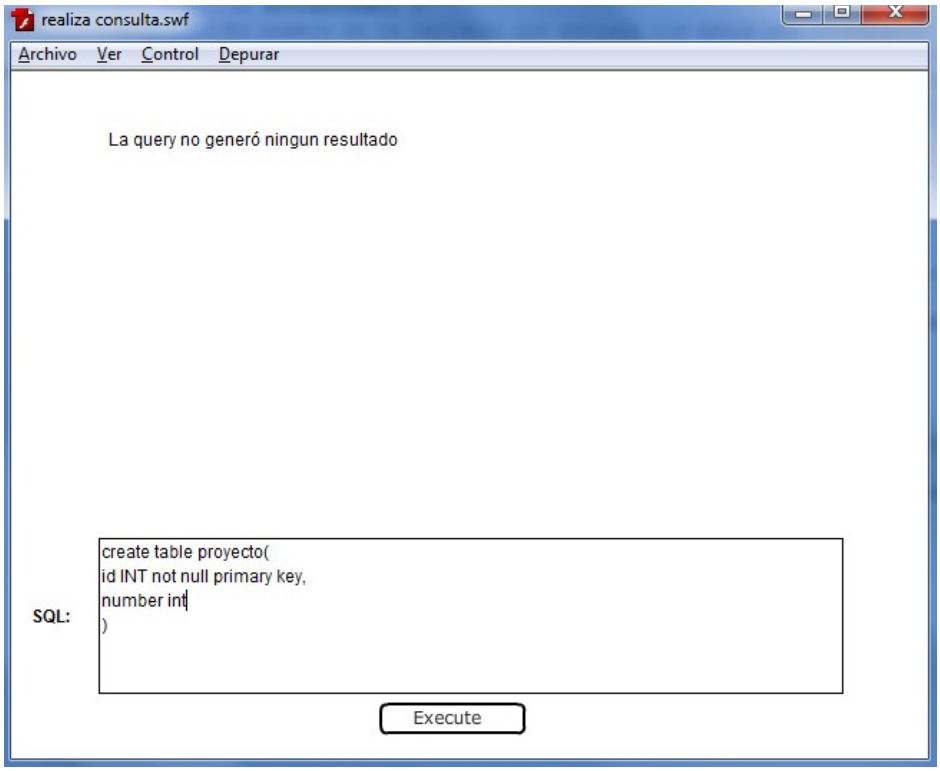


Figura 4.2.1.5. Flash, BD local: Modo de Empleo 1

Realizar Inserciones:

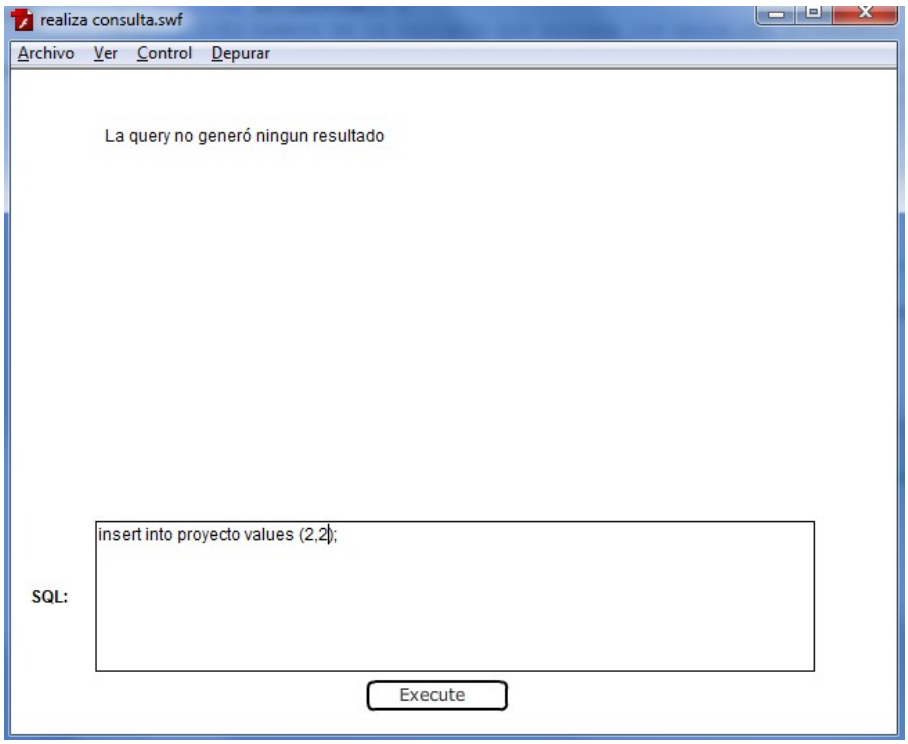


Figura 4.2.1.6. Flash, BD local: Modo de Empleo 2

O ejecutar consultas:

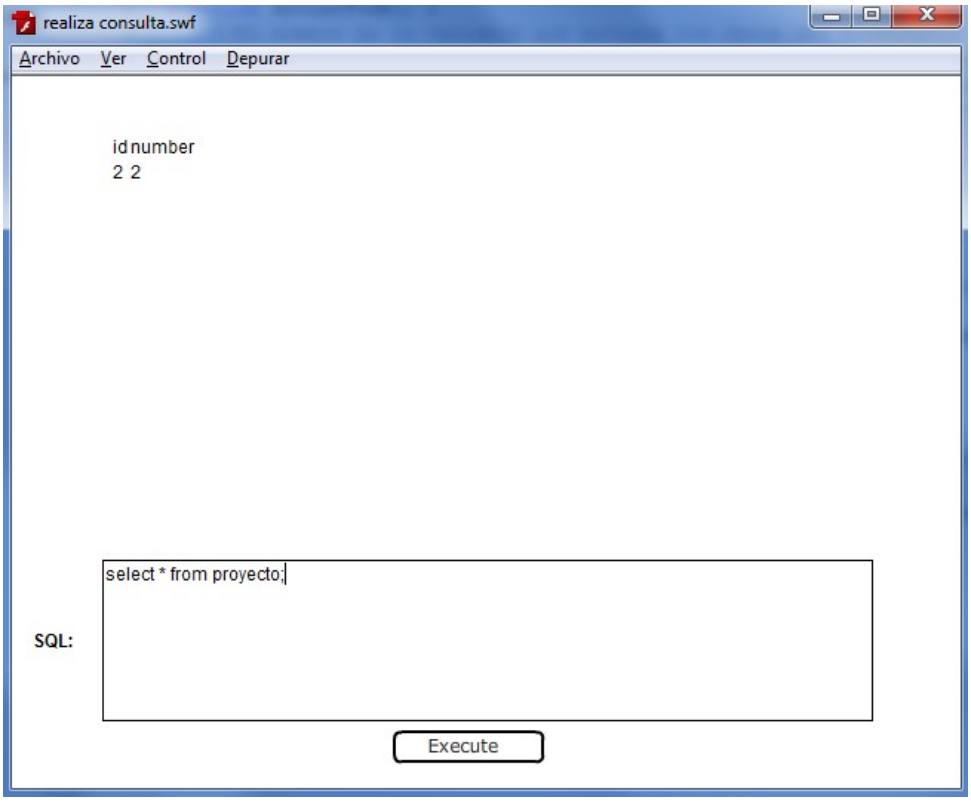


Figura 4.2.1.7. Flash, BD local: Modo de Empleo 3

Y si no encuentra el php:

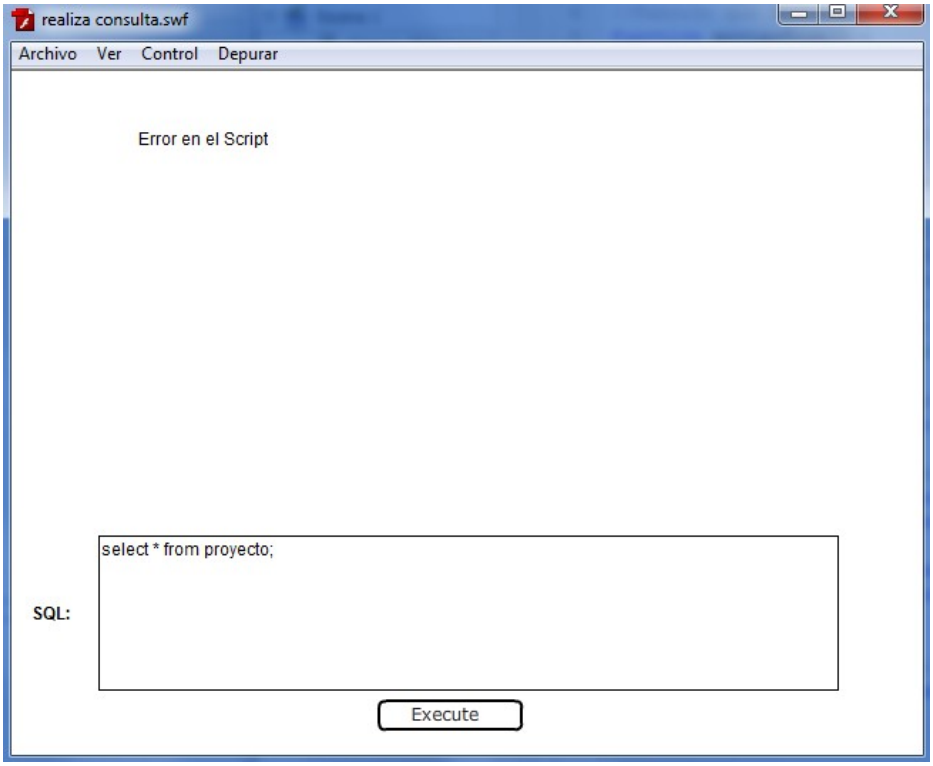


Figura 4.2.1.8. Flash, BD local: Modo de Empleo 4

4.3 Observaciones

Debido a que Macromedia Director MX 2004 está orientado a aplicaciones escritorio, la búsqueda de Xtras para acceso a base de datos ha sido larga y tediosa. Antes de decidirme por el Xtra xmysql, había empleado otros Xtras que me ofrecían funcionalidades parecidas. La cualidad que buscaba en estos Xtras era la de intentar lograr que la aplicación pudiera establecer una conexión a una base de datos remota, pero en ningún momento lo logré. Los Xtras que probé para esta finalidad fueron: ADOxtra, Arca y XmySQL.

A diferencia que Macromedia Director, Adobe Flash sí que proporciona un método para acceder a una base de datos remota. El problema que me encontré con Flash, fue que cuando quería establecer una conexión a una base de datos local, no tenía ninguna funcionalidad, a menos que quisiéramos montar la base de datos en Access.

Luego, las diferencias en este apartado sobre estas dos herramientas son:

- Director puede acceder a bases de datos locales sin la necesidad de un servidor
- Director tiene muchas dificultades para acceder a una base de datos remota
- Flash proporciona utilidades muy simples para acceder a bases de datos remotas, pero el inconveniente es que tienes que saber php o asp para ello.
- Flash no proporciona una manera directa para acceder a bases de datos locales.

Capítulo 5. APLICACIONES MULTIUSUARIO

En este último capítulo, se emplearán todos los componentes que hemos tratado para crear una aplicación multiusuario. La aplicación multiusuario que crearemos será un Chat. En el chat, nos tendremos que registrar para poder acceder, y una vez realizado el registro, se podrá chatear o ver el registro de mensajes de los usuarios.

5.1 Chat para Director

5.1.1 Distribución de los fotogramas

En el primer fotograma visualizaremos una pantalla de acceso, en la cual un usuario podrá darse de alta o registrarse, en función de la opción que seleccione:

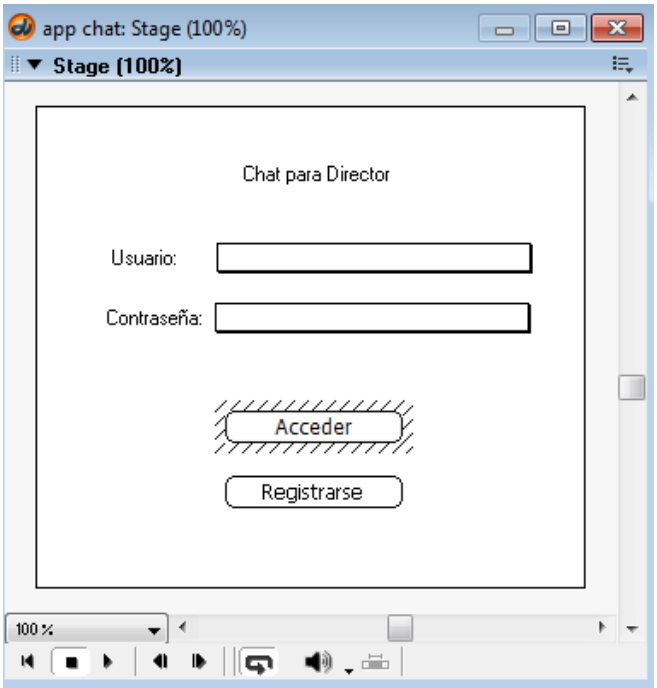


Figura 5.1.1.1. Director: Chat, Fotograma Inicial

Crearemos un segundo fotograma sobre el cual se visualizará el chat y en donde tendremos la opción de navegar a la consola de registro de mensajes de otros usuarios:

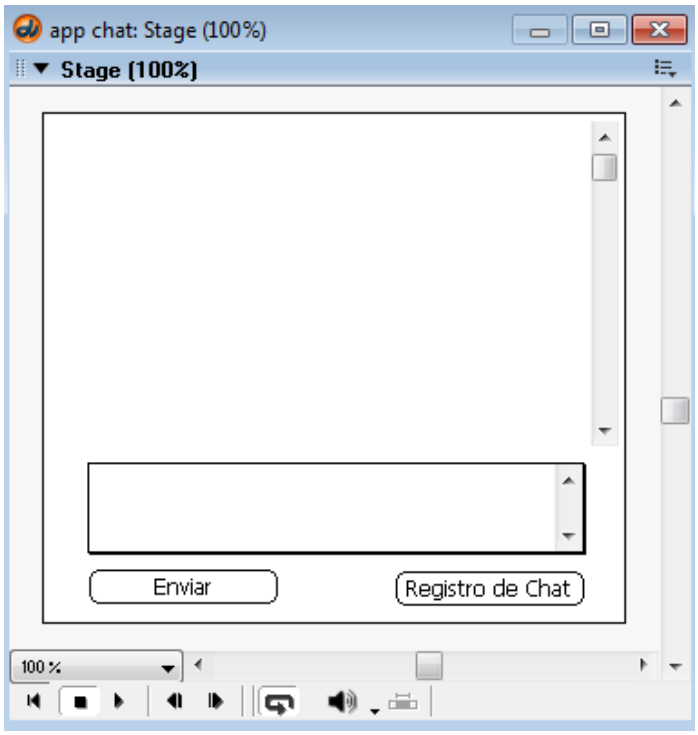


Figura 5.1.1.2. Director: Chat, Fotograma Chat

El tercer fotograma que crearemos representará la ventana de registro de chat, en la cual un usuario tendrá la opción de buscar todos los mensajes enviados por un usuario. También tendrá la opción de volver al chat:

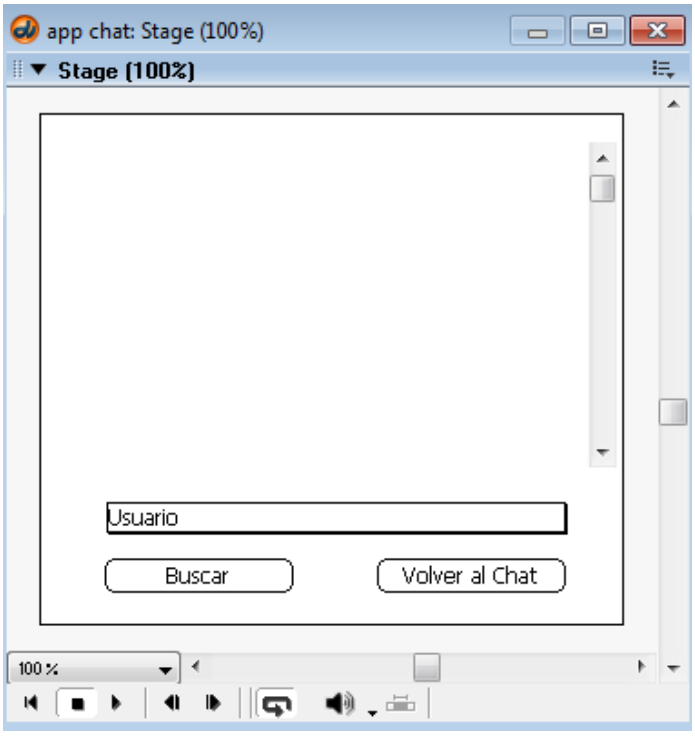


Figura 5.1.1.3. Director: Chat, Fotograma Registros

Por último, creamos un cuarto fotograma en el que se visualizará el un error en caso de no poder realizar la conexión al servidor:

Figura 5.1.1.4. Director: Chat, Fotograma Error

5.1.2 Scripts Lingo

5.1.2.1 Establecer conexión al iniciar la película

Para establecer la conexión inicial con el servidor, emplearemos el Xtra xmysql. Emplearemos un Script a nivel de Movie para establecer la conexión. En caso de no lograrlo, la película se situará sobre el fotograma de "Error de conexión":

```
global mysql,res
on startMovie
  global mysql
  set mysql= new (xtra "xmysql")
  if mysql.Xm_Connect("local-
host","root","root","chat",10111)=0 then
    set the text of member "ErrorText" to "No se pudo
establecer la conexión con el servidor"
    go to "Error"
    exit
  end if
  mysql.Xm_SelectDB("chat")
end startMovie
```

Como podemos observar, no comprobamos la existencia del servidor, ya que una aplicación servidor se encargará de lanzar el MySQL.

5.1.2.2 Realizar acceso de un usuario

Al pulsar el botón del primer fotograma “Acceder”, se verificarán el usuario y la contraseña introducidos con el contenido de la tabla “usuarios” de la base de datos. Si el acceso resulta ser satisfactorio, el usuario accederá al chat, en caso contrario, se mostrará un mensaje de error en la misma ventana, notificando del error:

```
on mouseUp me
  global mysql
  global user
  global lastMessageId
  lastMessageId = 0
  user = the text of member "user"
  password = the text of member "password"
  res = mysql.Xm_Query("SELECT * FROM usuarios where user =
'"&user&"' and pass = '"&password&"'")
  if res=0 then
    ErrMsg=mysql.Xm_GetError()
    if ErrMsg[1]="XmySQL-->No result set returned." then
      set the text of member "infoAcceso" to "Usuario o con-
traseña no válidos"
    else
      set the text of member "infoAcceso" to ErrMsg[2]
    end if
  else
    set the text of member "infoAcceso" to ""
    tmp = ""
    rows = mysql.Xm_RowCount(res)
    if rows>0 then
      set the text of member "infoAcceso" to ""
      mysql.Xm_FreeResult(res)
      res = mysql.Xm_Query("select max(id) from chat")
      rows = mysql.Xm_RowCount(res)
      if res>0 then
        rows = mysql.Xm_RowCount(res)
        if rows>0 then
          repeat with k = 1 to rows
            row=mysql.Xm_FetchRowList(res)
            lastMessageId = row[1]
          end repeat
        end if
        mysql.Xm_FreeResult(res)
      end if
      go to "Chat"
    else
      set the text of member "infoAcceso" to "Usuario o con-
traseña no válidos"
    end if
    mysql.Xm_FreeResult(res)
  end if
end
```

5.1.2.3 **Realizar registro de un usuario**

Al pulsar el botón del primer fotograma “Registro”, se verificará el usuario introducido con el contenido de la tabla “usuarios” de la base de datos. Si no se encuentra ninguna coincidencia, realizará un registro del usuario introducido con la contraseña especificada. Si el registro es satisfactorio o no, mostrará un mensaje indicando el resultado:

```
on mouseUp me
  global mysql
  user = the text of member "user"
  password = the text of member "password"
  res = mysql.Xm_Query("SELECT * FROM usuarios where user =
'"&user&"'")
  if res=0 then
    ErrMsg=mysql.Xm_GetError()
    if ErrMsg[1]="XmySQL-->No result set returned." then
      set the text of member "infoAcceso" to "El usuario
'"&user&"' se encuentra registrado. Intentalo con otro
nombre."
    else
      set the text of member "infoAcceso" to ErrMsg[2]
    end if
  else
    set the text of member "infoAcceso" to ""
    tmp = ""
    rows = mysql.Xm_RowCount(res)
    if rows>0 then
      set the text of member "infoAcceso" to "El usuario
'"&user&"' se encuentra registrado. Intentalo con otro
nombre."
    else
      mysql.Xm_FreeResult(res)
      res = mysql.Xm_Query("insert into usuarios values
('"&user&"','"&password&"');")
      set the text of member "infoAcceso" to "El usuario
'"&user&"' ha sido registrado satisfactoriamente!"
    end if
    mysql.Xm_FreeResult(res)
  end if
end
```

5.1.2.4 Enviar mensaje al Chat

En el segundo fotograma, al pulsar el botón “Enviar”, se almacenará el mensaje del usuario en la base de datos:

```
on mouseUp me
  global mysql
  global user
  content = the text of member "inputChat"
  res = mysql.Xm_Query("insert into chat (user, message) val-
ues ('"&user&"','"&content&"');")
```

```
  mysql.Xm_FreeResult(res)
  set the text of member "inputChat" to ""
end
```

5.1.2.5 Cargar contenido de la tabla Chat

Ahora que tenemos la funcionalidad de almacenar mensajes de Chat, necesitamos otra que constantemente nos proporcione los mensajes de chat nuevos que generen nuevas entradas.

Para ello, al comienzo de la ejecución del fotograma, realizaremos una búsqueda de nuevos mensajes en la entrada de chat y de existir nuevos mensajes, mostrarlos en el Chat:

```
on exitFrame me
  global mysql
  global lastMessageId
  res = mysql.Xm_Query("select * from chat where id>"&last-
MessageId)
  rows = mysql.Xm_RowCount(res)
  if res>0 then
    rows = mysql.Xm_RowCount(res)
    if rows>0 then
      repeat with k = 1 to rows
        row=mysql.Xm_FetchRowList(res)
        lastMessageId = row[1]
        set the text of member "chatResults" to the text of
member "chatResults"&""&row[2]&":"&tab&row[3]&return
      end repeat
    end if
    mysql.Xm_FreeResult(res)
  end if
  go to the frame
end
```

5.1.2.6 Acceder al registro de Chat

Para acceder al fotograma del registro de mensajes de otros usuarios, pulsaremos el botón del segundo fotograma “Registro de Chat”:

```
on mouseUp
   go to "Registro"
end
```

5.1.2.7 Volver a la ventana de Chat

Del mismo modo que accedemos al registro de Chat, debemos tener la opción de volver al Chat. Para ello pulsaremos el botón de “Volver al Chat” del tercer fotograma:

```
on mouseUp
   go to "Chat"
end
```

5.1.2.8 Ver registro de chat de un usuario

Para visualizar todos los mensajes que un usuario envió, pulsaremos el botón de "Buscar" en el tercer fotograma y realizaremos una búsqueda en la tabla de Chat y los mostraremos en el panel de resultados:

```
on mouseUp me
  global mysql
  user = the text of member "inputuser"
  password = the text of member "password"
  res = mysql.Xm_Query("SELECT * FROM chat where user =
'"&user&"'")
  if res=0 then
    ErrMsg=mysql.Xm_GetError()
    if ErrMsg[1]="XmySQL-->No result set returned." then
      set the text of member "userChatEntries" to "Usuario no
válido"
    else
      set the text of member "userChatEntries" to ErrMsg[2]
    end if
  else
    set the text of member "userChatEntries" to ""
    tmp = ""
    rows = mysql.Xm_RowCount(res)
    if rows>0 then
      textresult = ""
      repeat with k = 1 to rows
        row=mysql.Xm_FetchRowList(res)
        textresult = textresult&row[3]&return
      end repeat
      set the text of member "userChatEntries" to textresult
    else
      set the text of member "userChatEntries" to "Usuario no
válido"
    end if
    mysql.Xm_FreeResult(res)
  end if
end
```

5.1.3 Aplicación Servidor

Como aplicación servidor, emplearemos la aplicación para realizar comandos SQL de manera remota que creamos en el capítulo 4. De este modo, además de poder visualizar el contenido de las tablas, podemos definir el modelo de tablas que necesitamos para esta aplicación.

5.1.4 Tablas para la aplicación

El último paso es la creación de las tablas para la aplicación. Si hemos estado observando el código, veremos que necesitaremos la creación de dos tablas:

- La tabla "usuarios" en la que se almacenarán los distintos usuarios:

```
CREATE TABLE usuarios(
     user VARCHAR(15) NOT NULL PRIMARY KEY,
     pass VARCHAR(15) NOT NULL
)
```

- La tabla "chat" para almacenar los mensajes que enviaron los usuarios:

```
CREATE TABLE chat(
     id INT(10) NOT NULL AUTO_INCREMENT PRIMARY KEY,
     user VARCHAR(15) NOT NULL,
     message VARCHAR(200) NOT NULL,
     CONSTRAINT PRIMARY KEY (user) REFERENCES usuarios (user)
)
```

5.1.5 Modo de empleo de la aplicación

Inicialmente, arrancaremos la aplicación servidor del cuarto capítulo:

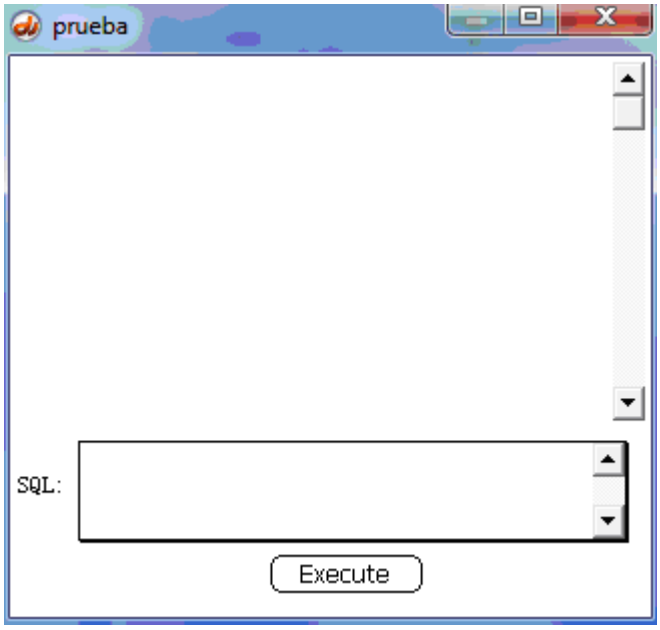


Figura 5.1.5.1. Director: Chat, Modo de empleo 1

A continuación, o bien abrimos dos ventanas de chat en nuestro ordenador, o bien realizamos una conexión en nuestro ordenador y otro desde la misma red de área local.

Una vez iniciadas las dos ventanas de chat, el siguiente paso será registrarse:

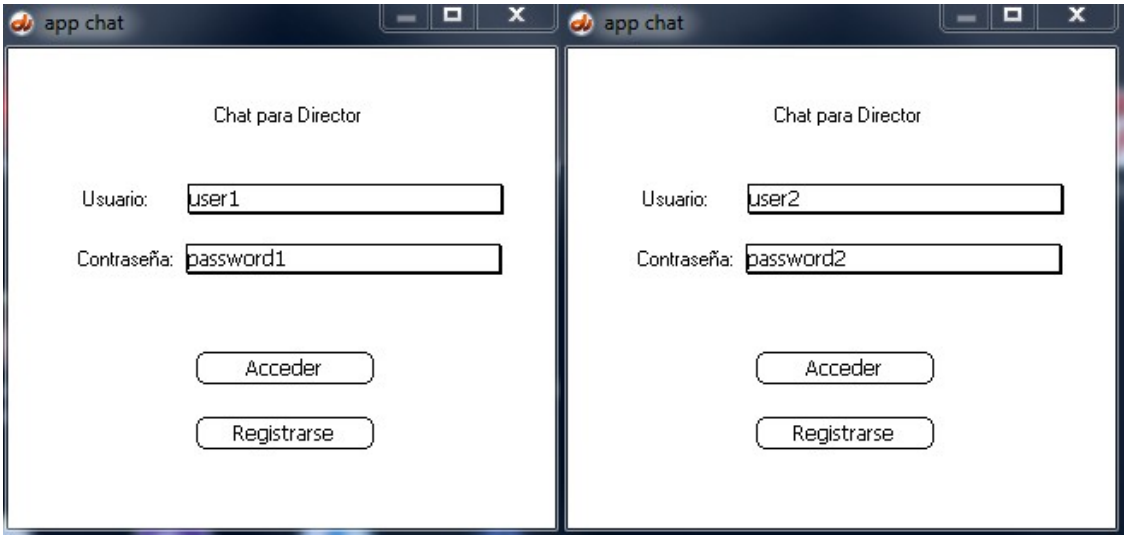


Figura 5.1.5.2. Director: Chat, Modo de empleo 2

Ahora ya podemos emplear el chat:

Figura 5.1.5.3. Director: Chat, Modo de empleo 3

O ver si algún usuario ha comentado algo en nuestra ausencia:

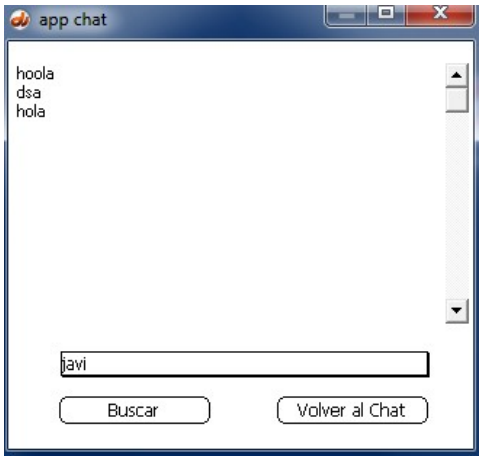


Figura 5.1.5.4. Director: Chat, Modo de empleo 4

5.2 Chat para Flash

5.2.1 Distribución de los fotogramas

Crearemos una capa en la que visualizaremos los fotogramas de la película. En el primer fotograma clave se mostrará la pantalla de acceso, en la cual, al igual que en Director, un usuario podrá darse de alta o registrarse:

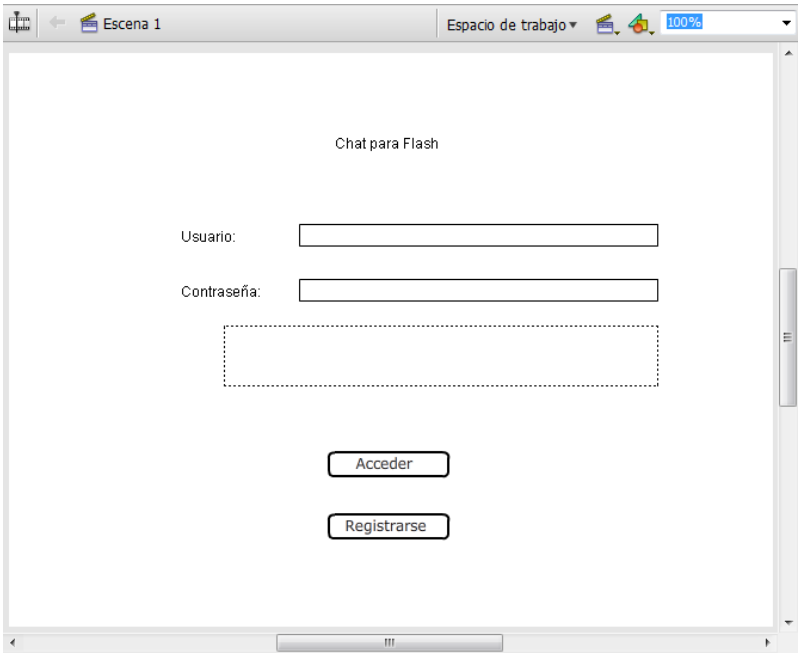


Figura 5.2.1.1. Flash: Chat, Fotograma Inicial

Crearemos un segundo fotograma clave sobre el cual se visualizará el Chat. Se podrá navegar a la consola de registro de mensajes de otros usuarios:

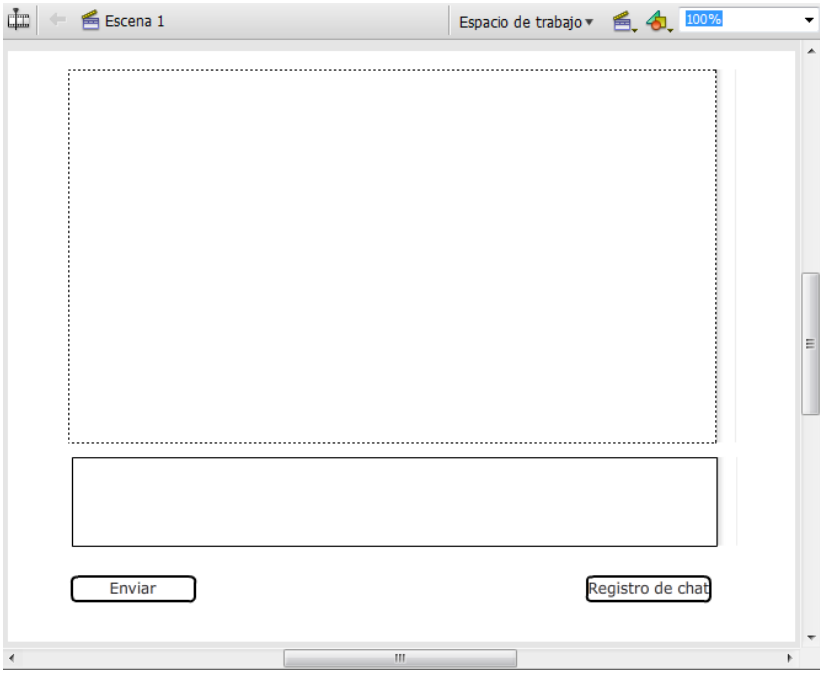


Figura 5.2.1.2. Flash: Chat, Fotograma Chat

El tercer fotograma clave que crearemos representará la ventana de registro de Chat, en la cual un usuario tendrá la opción de buscar todos los mensajes enviados por un usuario. También tendrá la opción de volver al Chat:

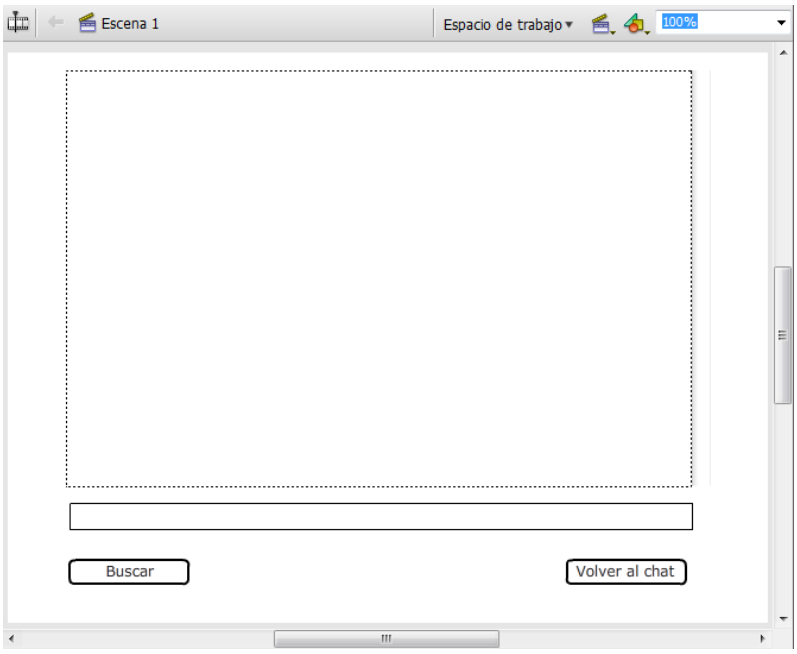


Figura 5.2.1.3. Flash: Chat, Fotograma Registros

5.2.2 Scripts Action Script y php

Además de la parte Flash, debemos programar en el lado del servidor los php oportunos para que las funciones que programemos tengan sentido.

También emplearemos dos variables globales para manejar información de un fotograma a otro:

```
var userName;
var lastMessageId;
```

5.2.2.1 Realizar acceso de un usuario

Al pulsar el botón del primer fotograma "Acceder", se verificarán el usuario y la contraseña introducidos con el contenido de la tabla "usuarios" de la base de datos. Si el acceso resulta ser satisfactorio, el usuario accederá al chat, en caso contrario, se mostrará un mensaje de error en la misma ventana, notificando del error:

Script del botón "Acceder":

```
on (release) {
     acceder();
}
```

Script función acceder():

```
//variables para enviar y para recibir información
var envio_acceso: LoadVars = new LoadVars();
var recibir_acceso: LoadVars = new LoadVars();
//Función que envia el formulario de acceso
function acceder() {
     envio_acceso.user = user_input.text;
     envio_acceso.pass = pass_input.text;
     envio_acceso.sendAndLoad( "http://localhost/chat/acceso.
     php", recibir_acceso, "POST");
}
//Función que procesa los datos recibidos del servidor
recibir_acceso.onLoad = function (exito) {
     if (exito) {
          if(this.mensaje ==undefined || this.mensaje == ""){
               mensageInfo.text = "Usuario o contraseña no
válidos.";
          } else {
               if(this.mensaje=="ok"){
                    userName = user_input.text;
                    getLastId();
                    gotoAndStop(2);
               } else {
                    mensageInfo.text = this.mensaje;
               }
               this.mensaje = undefined;
          }
     } else {
          mensageInfo.text = "Error en el Script";
     }
};
```

acceso.php:

```
<?
     $chatUser=$_POST["user"];
     $chatPass=$_POST["pass"];
     $host = "localhost";
     $user = "root";
     $pass = "root";
     $bbdd = "chat";
     $consulta = "SELECT * FROM usuarios WHERE user =
'".$chatUser."' AND pass = '".$chatPass."' ";
     $conexion = mysql_connect($host,$user,$pass) or
die("mensaje=".mysql_error());
     mysql_select_db($bbdd,$conexion) or
die("mensaje=".mysql_error());
     $result = mysql_query($consulta,$conexion) or
die("mensaje=".mysql_error());
     $respuesta = "";
     $numfield = 0;
     if($val=mysql_fetch_array($result)) {
          $numfield++;
```

```
                $respuesta = "ok";
            }
            if($numfield==0){
              $respuesta = $respuesta . "Usuario o contraseña
no válidos.\n";
            }
            echo "mensaje=".$respuesta."";
      ?>
```

5.2.2.2 **Realizar registro de un usuario**

Al pulsar el botón del primer fotograma "Registro", se verificará el usuario introducido con el contenido de la tabla "usuarios" de la base de datos. Si no se encuentra ninguna coincidencia, realizará un registro del usuario introducido con la contraseña especificada. Si el registro es satisfactorio o no, mostrará un mensaje indicando el resultado:

Script del botón "Registrarse":

```
on (release) {
      registrar();
}
```

Script de la función registrar():

```
//variables para enviar y para recibir información
var envio_registro: LoadVars = new LoadVars();
var recibir_registro: LoadVars = new LoadVars();
//Función que envía el formulario de registro
function registrar() {
      envio_registro.user = user_input.text;
      envio_registro.pass = pass_input.text;
      envio_registro.sendAndLoad( "http://localhost/chat/regis
      tro.php", recibir_registro, "POST");
}
//Función que procesa los datos recibidos del servidor
recibir_registro.onLoad = function (exito) {
      if (exito) {
            if(this.mensaje ==undefined || this.mensaje == ""){
                  mensageInfo.text = "El usuario
                  '"+user_input.text+"' ha sido registrado
                  satisfactoriamente!";
            } else {
                  mensageInfo.text = this.mensaje;
                  this.mensaje = undefined;
            }
      } else {
            mensageInfo.text = "Error en el Script";
      }
};
```

registro.php:

```
<?
	$chatUser=$_POST["user"];
	$chatPass=$_POST["pass"];
	$host = "localhost";
	$user = "root";
	$pass = "root";
	$bbdd = "chat";
	$consulta = "INSERT INTO usuarios (user,pass) VALUES
('".$chatUser."','".$chatPass."') ";
	$conexion = mysql_connect($host,$user,$pass) or
die("mensaje=".mysql_error());
	mysql_select_db($bbdd,$conexion) or
die("mensaje=".mysql_error());
	$result = mysql_query($consulta,$conexion) or
die("mensaje=El usuario '".$chatUser."' se encuentra
registrado. Intentalo con otro nombre.");
?>
```

5.2.2.3 Enviar mensaje al Chat

En el segundo fotograma, al pulsar el botón "Enviar", se almacenará el mensaje del usuario en la base de datos:

Script del botón "Enviar":

```
on (release) {
	sendMessage();
}
```

Script de la función sendMessage():

```
//variables para enviar y para recibir información
var envio_message: LoadVars = new LoadVars();
var recibir_message: LoadVars = new LoadVars();
//Función que envía el mensaje introducido por el usuario
function sendMessage() {
	envio_message.user = userName;
	envio_message.mssg = messageInput.text;
	envio_message.sendAndLoad( "http://localhost/chat/sendCh
	at.php", recibir_message, "POST");
	messageInput.text = "";
}
```

sendChat.php:

```
<?
	$chatUser=$_POST["user"];
	$chatMessage=$_POST["mssg"];
```

```
     $host = "localhost";
     $user = "root";
     $pass = "root";
     $bbdd = "chat";
     $consulta = "INSERT INTO chat (user,message) VALUES ('".
$chatUser."','".$chatMessage."') ";
     $conexion = mysql_connect($host,$user,$pass) or
die("mensaje=".mysql_error());
     mysql_select_db($bbdd,$conexion) or
die("mensaje=".mysql_error());
     $result = mysql_query($consulta,$conexion) or
die("mensaje=".mysql_error());
?>
```

5.2.2.4 Cargar contenido de la tabla Chat

Ahora que tenemos la funcionalidad de almacenar mensajes de Chat, necesitamos otra que constantemente nos proporcione los mensajes de Chat nuevos que generen nuevas entradas.

Para ello, al comienzo de la ejecución del fotograma, realizaremos una búsqueda de nuevos mensajes en la entrada de Chat y de existir nuevos mensajes, mostrarlos en el Chat. Cuando termine la búsqueda, volveremos a realizar otra, de esta manera hasta el fin de la ejecución de la película. Además, emplearemos una función para obtener el identificador de la última entrada del chat, en caso de que no lo sepamos:

Script para la función cargarChat():

```
//Función para cargar el contenido del chat
function cargarChat() {
     envio_contenidoChat.id = lastMessageId;
     envio_contenidoChat.sendAndLoad( "http://localhost/chat/
     contenidoChat.php", recibir_contenidoChata, "POST");
}
//Función que carga el contenido del chat en el chat
recibir_contenidoChata.onLoad = function (exito) {
     if (exito) {
          if(this.mensaje !=undefined && this.mensaje != ""){
               chatOutput.text = chatOutput.text + "" +
this.mensaje;
               this.mensaje = undefined;
               lastMessageId = this.lastId;
               //puede ser que alguien ha
               this.lastId = undefined;
          }
          cargarChat();
     } else {
          chatOutput.text = "Error en el Script";
     }
};
```

contenidoChat.php:

```
<?
     $id = $_POST["id"];
     if($id!='undefined'){
          $consulta = "SELECT * FROM chat WHERE id>".$id." ";
          $host = "localhost";
          $user = "root";
          $pass = "root";
          $bbdd = "chat";
          $conexion = mysql_connect($host,$user,$pass) or
die("mensaje=".mysql_error());
          mysql_select_db($bbdd,$conexion) or
die("mensaje=".mysql_error());
          $result = mysql_query($consulta,$conexion) or
die("mensaje=".mysql_error());
          $respuesta = "";
          $lastId = "";
          while($val=mysql_fetch_array($result)) {
               $respuesta = $respuesta."".$val[1].":\t".
$val[2]."\n";
               $lastId = $val[0];
          }
          echo "mensaje=".$respuesta."";
          echo "&lastId=".$lastId;
     }
?>
```

Script para la función getLastId():

```
//variables para enviar y para recibir información
var envio_peticionId: LoadVars = new LoadVars();
var recibir_peticionId: LoadVars = new LoadVars();
//Función para obtener el último id de la base de datos
function getLastId(){
     envio_peticionId.sendAndLoad( "http://localhost/chat/get
     LastChatId.php", recibir_peticionId, "POST");
}
//Función que carga el contenido del chat en el chat
recibir_peticionId.onLoad = function (exito) {
     if (exito) {
          if(this.mensaje !=undefined && this.mensaje != ""){
               lastMessageId = this.mensaje;
               this.mensaje = undefined;
          } else {
               lastMessageId = "0";
          }
     } else {
          lastMessageId = "0";
     }
};
```

getLastChatId.php:

```
<?
      $host = "localhost";
      $user = "root";
      $pass = "root";
      $bbdd = "chat";
      $consulta = "SELECT MAX(id) FROM chat ";
      $conexion = mysql_connect($host,$user,$pass) or
die("mensaje=".mysql_error());
      mysql_select_db($bbdd,$conexion) or
die("mensaje=".mysql_error());
      $result = mysql_query($consulta,$conexion) or
die("mensaje=".mysql_error());
      $respuesta = "";
      $numfield = 0;
      if($val=mysql_fetch_array($result)) {
            $respuesta = $val[0];
            $numfield++;
      }
      echo "mensaje=".$respuesta."";
?>
```

5.2.2.5 Acceder al registro de Chat

Para acceder al fotograma del registro de mensajes de otros usuarios, pulsaremos el botón del segundo fotograma “Registro de Chat”:

```
on (release) {
      gotoAndStop(3);
}
```

5.2.2.6 Volver a la ventana de Chat

Del mismo modo que accedemos al registro de Chat, debemos tener la opción de volver al Chat. Para ello pulsaremos el botón de “Volver al Chat” del tercer fotograma:

```
on (release) {
      gotoAndStop(2);
}
```

5.2.2.7 Ver registro de Chat de un usuario

Para visualizar todos los mensajes que un usuario envió, pulsaremos el botón de “Buscar” en el tercer fotograma y realizaremos una búsqueda en la tabla de Chat y los mostraremos en el panel de resultados:

Script del botón “Buscar”:

```
on (release) {
      buscarRegistroDeUsuario();
}
```

Script de la función buscarRegistroDeUsuriario():

```
//variables para enviar y para recibir información
var envio_peticionDeRegistro:LoadVars = new LoadVars();
var recibir_peticionDeRegistro:LoadVars = new LoadVars();
//Función para obtener toda la conversación de un usuario
function buscarRegistroDeUsuario(){
     envio_peticionDeRegistro.findUser = findUserInput.text;
     envio_peticionDeRegistro.sendAndLoad( "http://localhost/
     chat/registroUsuario.php", recibir_peticionDeRegistro,
     "POST");
}
//Función que carga el contenido del chat de un usuario
recibir_peticionDeRegistro.onLoad = function (exito) {
     if (exito) {
          if(this.mensaje !=undefined && this.mensaje != ""){
               outputUserChat.text = this.mensaje;
               this.mensaje = undefined;
          } else {
               outputUserChat.text = "El usuario que has
introducido no existe o no ha escrito nada.";
          }
     } else {
          chatOutput.text = "Error en el Script";
     }
}
```

registroUsuario.php:

```
<?
     $findUser = $_POST["findUser"];
     $host = "localhost";
     $user = "root";
     $pass = "root";
     $bbdd = "chat";
     $consulta = "SELECT * FROM chat WHERE user = '".
$findUser."' ";
     $conexion = mysql_connect($host,$user,$pass) or
die("mensaje=".mysql_error());
     mysql_select_db($bbdd,$conexion) or
die("mensaje=".mysql_error());
     $result = mysql_query($consulta,$conexion) or
die("mensaje=".mysql_error());
     $respuesta = "";
     $lastId = "";
     while($val=mysql_fetch_array($result)) {
          $respuesta = $respuesta . "" . $val[2] . "\n";
          $lastId = $val[0];
     }
     echo "mensaje=".$respuesta."";
?>
```

5.2.3 Tablas para la aplicación

El último paso es la creación de las tablas para la aplicación. Como en Director, necesitaremos dos tablas para el correcto empleo de la aplicación:

- La tabla “usuarios” en la que se almacenarán los distintos usuarios:

```
CREATE TABLE usuarios(
     user VARCHAR(15) NOT NULL PRIMARY KEY,
     pass VARCHAR(15) NOT NULL
)
```

- La tabla “chat” para almacenar los mensajes que enviaron los usuarios:

```
CREATE TABLE chat(
     id INT(10) NOT NULL AUTO_INCREMENT PRIMARY KEY,
     user VARCHAR(15) NOT NULL,
     message VARCHAR(200) NOT NULL,
     CONSTRAINT PRIMARY KEY (user) REFERENCES usuarios (user)
)
```

5.2.4 Modo de empleo de la aplicación

Una vez publicada la aplicación, la podemos almacenar en el servidor para acceder a ello a través de la URL: http://urlservidor/localizaciónAplicación/chat.html:

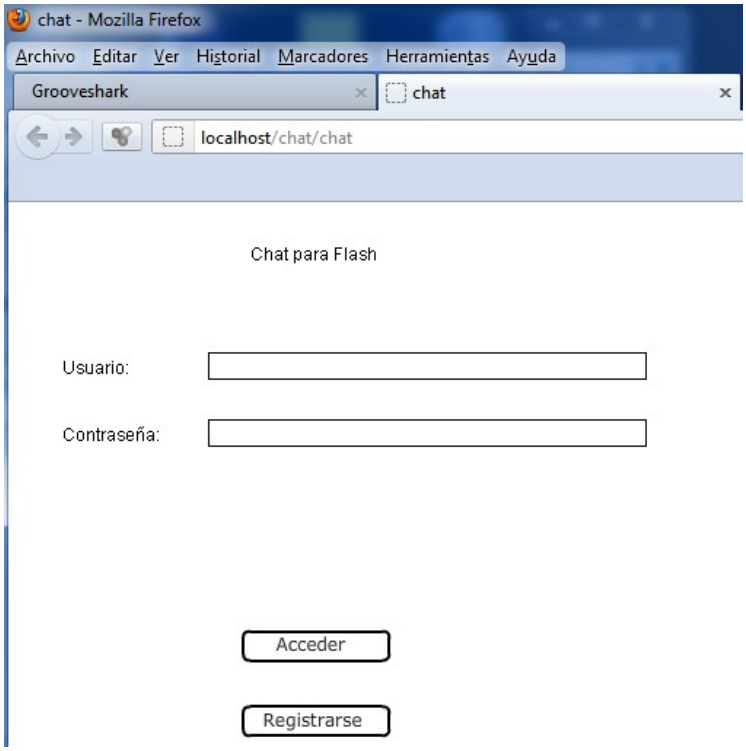


Figura 5.2.4.1. Flash: Chat, Modo de empleo 1

Una vez iniciada la ventana de chat, el siguiente paso será registrarse:

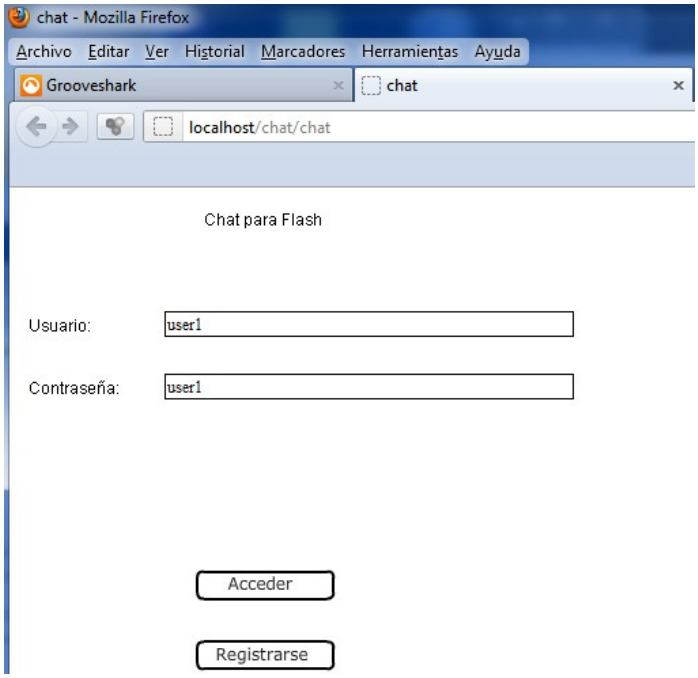


Figura 5.2.4.2. Flash: Chat, Modo de empleo 2

Ahora ya podemos emplear el chat:

Figura 5.2.4.3. Flash: Chat, Modo de empleo 3

O ver si algún usuario ha comentado algo en nuestra ausencia:

Figura 5.2.4.4. Flash: Chat, Modo de empleo 4

5.3 Observaciones

Como hemos podido apreciar, para crear un chat en Director como en Flash, hemos necesitado una elaboración más profunda. Sin embargo, la creación de la aplicación ha sido mucho más sencilla en Director que en Flash, ya que en ningún momento hemos tenido la necesidad de interactuar con una capa externa para que realice las peticiones generadas. En cambio, el Chat de Director únicamente se podría emplear en redes de área local, mientra que el Chat de Flash se podría emplear con cualquier conexión a la red que tuviera acceso al servidor.

Capítulo 6. CONCLUSIONES

La continua elaboración de las aplicaciones y el análisis de las herramientas nos han permitido obtener experiencia e información sobre estas herramientas y hemos podido ver sus diferentes puntos fuertes y débiles.

Cuando creamos nuestra primera aplicación, en un principio todo parecían ventajas para la herramienta Director, ya que nos agilitaba mucho la generación y manipulación de los elementos. Pero no todo eran ventajas, ya que pudimos observar que la película resultante que generaba Director frente a Flash ocupaba en espacio de disco una cantidad significativamente superior.

Al comenzar con la manipulación de las bases de datos, ya empezamos a observar las limitaciones de la herramienta Director frente a Flash, viendo que se necesitaban emplear plugins externos para emplear nuevas funcionalidades mientras que Flash nos proporcionaba herramientas para la obtener la comunicación necesaria.

Por último, para la aplicación multiusuario de chat, comprobamos que ambas herramientas permitían asociar código a un nivel más complejo. Una vez más, las limitaciones de Director frente a Flash para la comunicación en la red se volvieron a dar, provocando que el chat para Director únicamente pudiera funcionar en redes internas, mientras que el chat para Flash podía funcionar sobre la red.

En mi opinión, Director es una buena herramienta para introducirse a las herramientas de autor, para aquellos desarrolladores que no tengan experiencia en esta materia. En caso contrario, si un usuario está familiarizado con la manipulación de herramientas autor, Flash nos ofrece una variedad más rica y compleja para trabajar. Sus continuas actualizaciones hacen posible generar aplicaciones que se demandan en la actualidad.

Capítulo 7. REFERENCIAS

ENLACES:

http://ima.udg.edu/~gonzalo/teaching/IntroduccionLingo.pdf
http://www.xtra-ucd.com/
http://www.appservnetwork.com/

LIBROS:

3D for the Web (Carol MacGillivray, Anthony Head)
Programming ActionScript 2.0 (Adobe)
Macromedia Director MX 2004 (Macromedia)

HERRAMIENTAS DE AUTOR PARA DESARROLLO DE APLICACIONES MULTIMEDIA INTERACTIVAS MULTIUSUARIO: ANÁLISIS Y RECOMENDACIONES

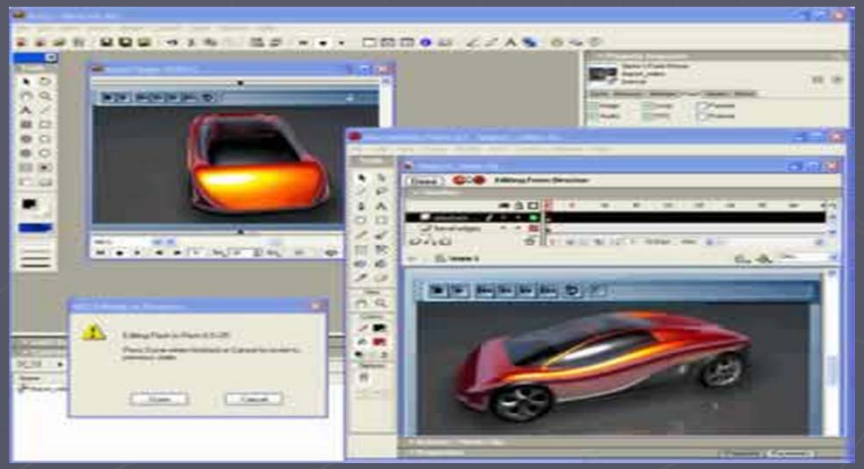
Javier Jorge Soteras

Objeto:

- Guiar a un usuario final por las herramientas autor y ser capaz de evolucionar de la herramienta Director a la herramienta Flash

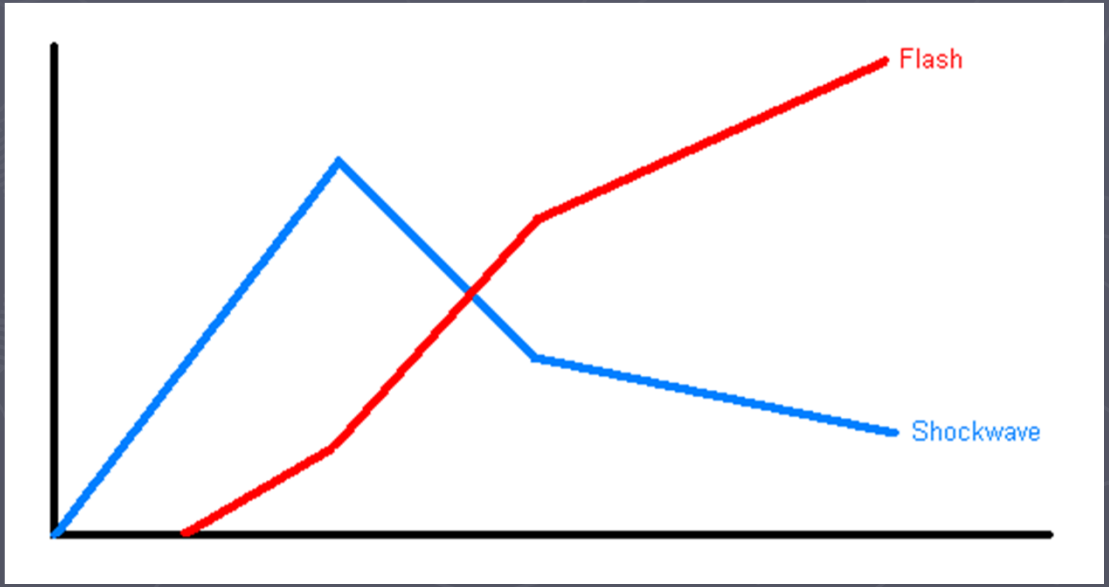
Herramientas

- Herramientas Autor:
  - Director MX 2004
  - Adobe Flash CS3
- Lenguajes Autor:
  - Lingo
  - Action Script 2



Evolución

- Director nació en 1985 orientado a aplicaciones digitales
- Flash nació en 1996 orientado a la Web
- Actualmente Shockwave está instalado en un 50% de los navegadores, mientras que Flash está presente en el 98%

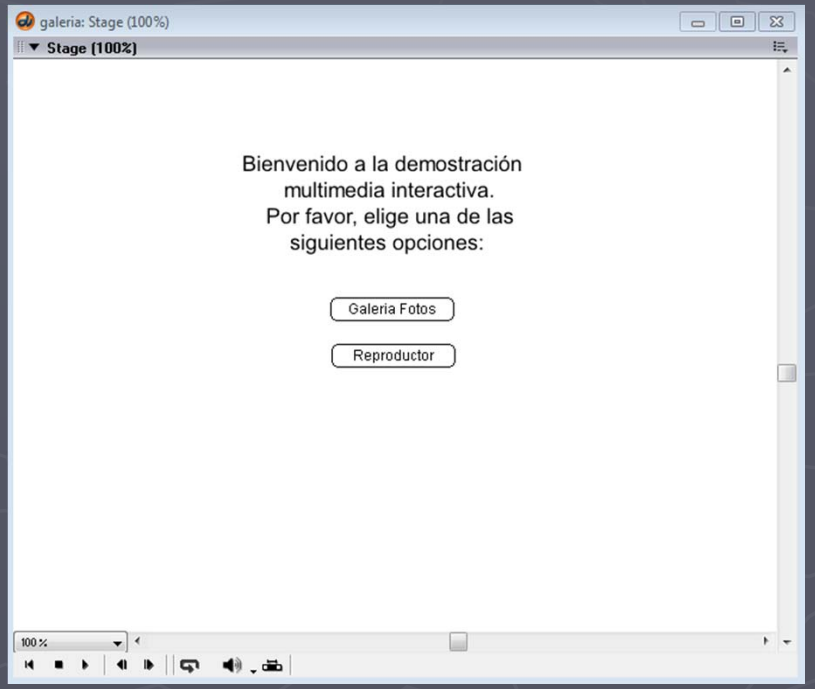


Componentes

Visualización

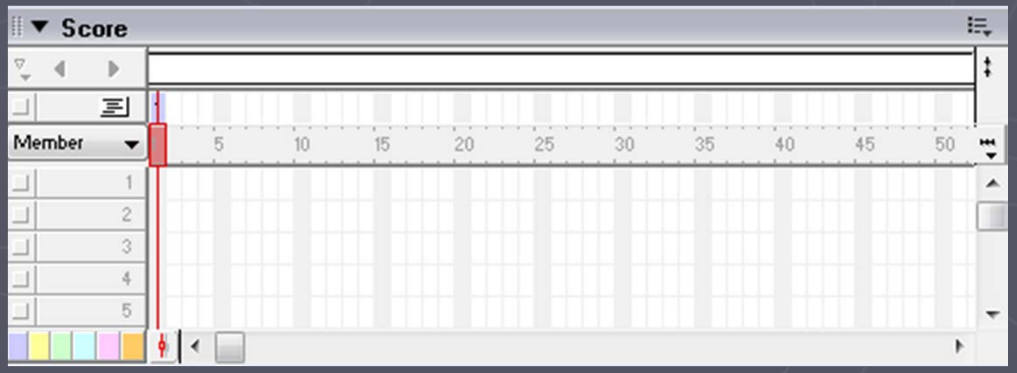
- Director
  - Ventana Stage
- Flash
  - Ventana de Escena

En ambos casos tendremos la opción de visualizar el contenido de la película en función del fotograma sobre el que nos situamos



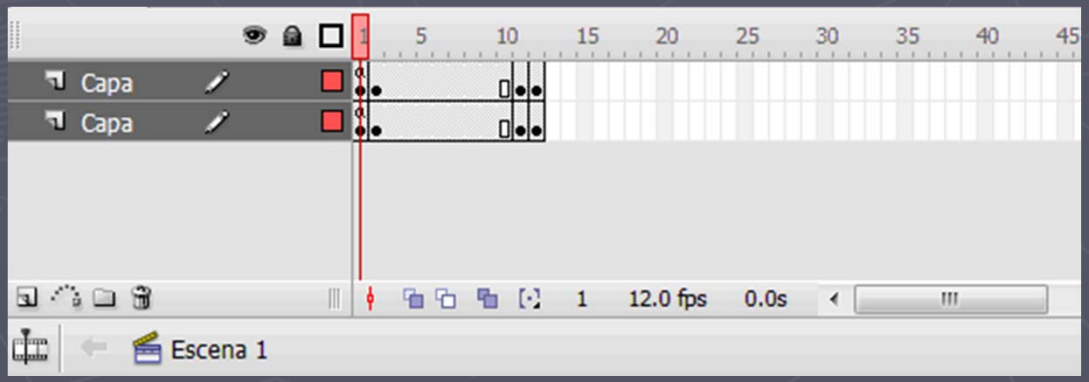
Manipulación de la línea del tiempo y sus elementos

- Director
  - Ventana Score
    - Sprites: Los elementos que participarán en la película
    - Fotogramas: Unidad métrica de la película



Manipulación de la línea del tiempo y sus elementos

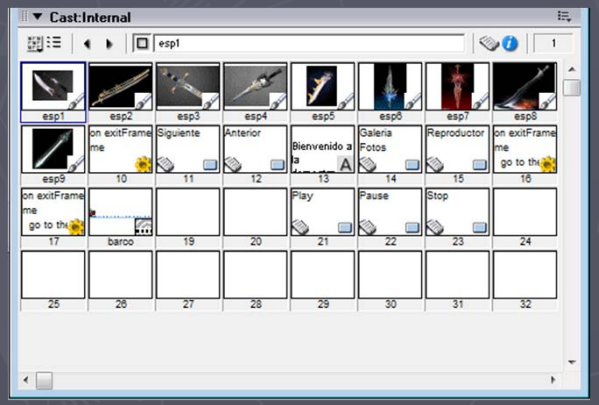
- Flash
  - Línea del tiempo
    - Capas: Representación de los Sprites. En una capa pueden existir distintos elementos.
    - Fotogramas: Unidad métrica de la película.

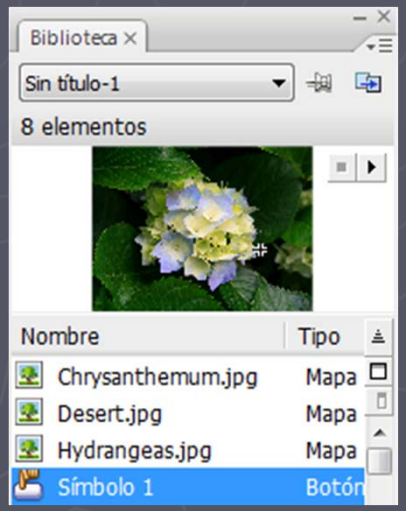


Contenedor de Objetos

- Director
  - Ventana Cast
- Flash
  - Biblioteca

Ubicación de los elementos multimedia (Imágenes, videos, audio, ...)

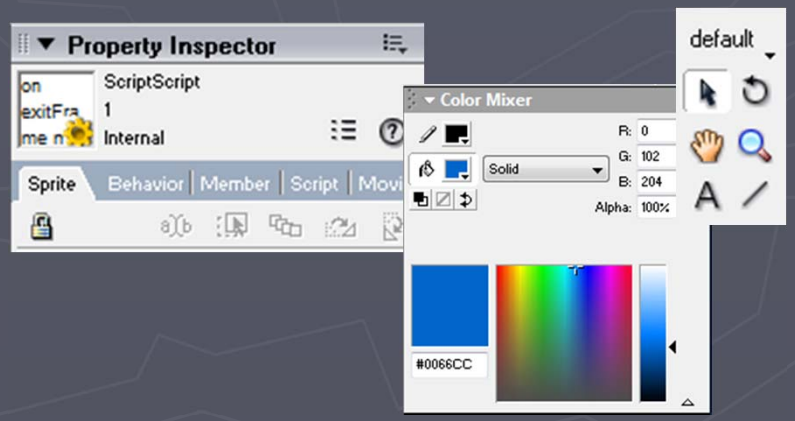




Otras herramientas

Ambos contienen diversas herramientas para la manipulación de los elementos

- Director
  - Barra de herramientas
  - Inspectores
- Flash
  - Barra de herramientas
  - Paneles



Lenguaje Autor

Lenguaje Autor

Lenguaje orientado al desarrollo de elementos multimedia

Nos aportarán la capacidad de manipular elementos multimedia en las respectivas herramientas

- Director
  - Lingo
- Flash
  - Action Script

Uso de las herramientas autor: Aplicación Interactiva

Aplicación Interactiva

- Introducción a los elementos multimedia
- Primeros pasos con la programación autor
- Manejo de objetos multimedia
- Empleo de la línea del tiempo
- Asignación de scripts sobre objetos multimedia
- Generar una película ejecutable

Aplicación interactiva

- Para tratar todos estos puntos:

  GALERÍA-REPRODUCTOR

Aplicación Interactiva: Galería

- Director
  - Importar Cast members (imágenes y película)
  - Configuración de la ventana Score
  - Poblar fotogramas
  - Insertar código Lingo
  - Generar película
- Flash
  - Añadir elementos a la biblioteca
  - Configurar capas de la película
  - Poblar fotogramas
  - Insertar código AS
  - Generar película

Aplicación Interactiva: Galería

- Director
  - Simple y rápido de manejar
  - Requiere definir todos los elementos a emplear
  - Aplicaciones ocupan mucho espacio en disco
- Flash
  - Desarrollo de elementos más elaborados
  - No requiere definir todos los elementos a emplear
  - Aplicaciones ligeras. Perfectas para la red

Manejo de bases de datos sobre las herramientas autor

Manejo de bases de datos sobre las herramientas autor

- Director
  - No facilita ninguna herramienta para el acceso a bases de datos
  - Xtra XMySQL: plugin creado por programador externo
  - Inconveniente: No accede a bases de datos externas
- Flash
  - Proporciona herramienta para realizar peticiones GET y POST → LoadVars
  - Inconveniente: Requiere la presencia de un servidor (externo o interno)

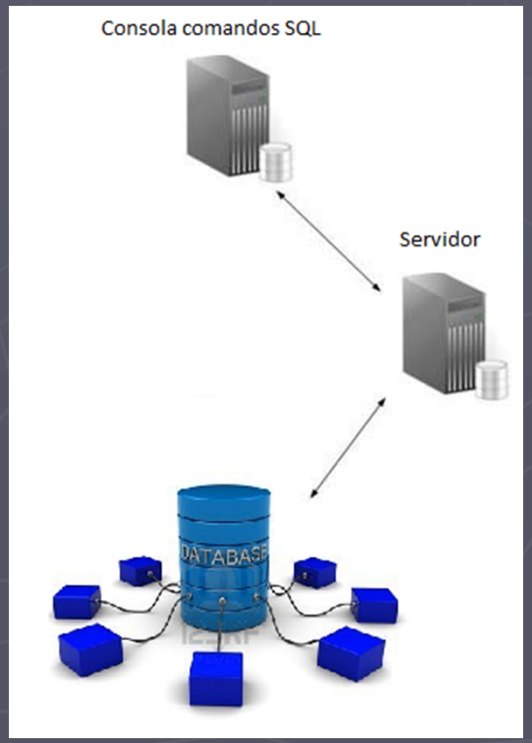
Manejo de bases de datos sobre las herramientas autor

- Para ilustrar estas funcionalidades crearemos una consola de comandos SQL
  - Director
    - La aplicación consultará directamente sobre el servidor MySQL
  - Flash
    - Requiere la presencia de un servidor (externo o interno) que realice las peticiones al servidor MySQL

Manejo de bases de datos sobre las herramientas autor: Consola de comandos SQL

- Director

- Flash



Manejo de bases de datos sobre las herramientas autor: Consola de comandos SQL

- Director
  - Empleo del API del Xtra XMySQL
  - Ejecución de consultas SQL mediante XMySQL
  - Control de status del servidor MySQL
- Flash
  - Montar servidor WAMP
  - Empleo del objeto LoadVars para realizar peticiones POST al servidor
  - Generar scripts PHP que ejecuten querys SQL sobre el servidor MySQL y devuelvan resultado obtenido

Manejo de bases de datos sobre las herramientas autor: Consola de comandos SQL

- Director
  - API sencillo para manejo completo del servidor MySQL
  - Imposibilidad de empleo fuera de la red
  - Requiere el mantenimiento del servidor MySQL
- Flash
  - Requiere servidor externo
  - No se encarga del mantenimiento del servidor MySQL
  - Las consultas no sobrecargan la aplicación, ya que se ejecutan sobre servidor, no sobre cliente

Aplicaciones Multiusuario

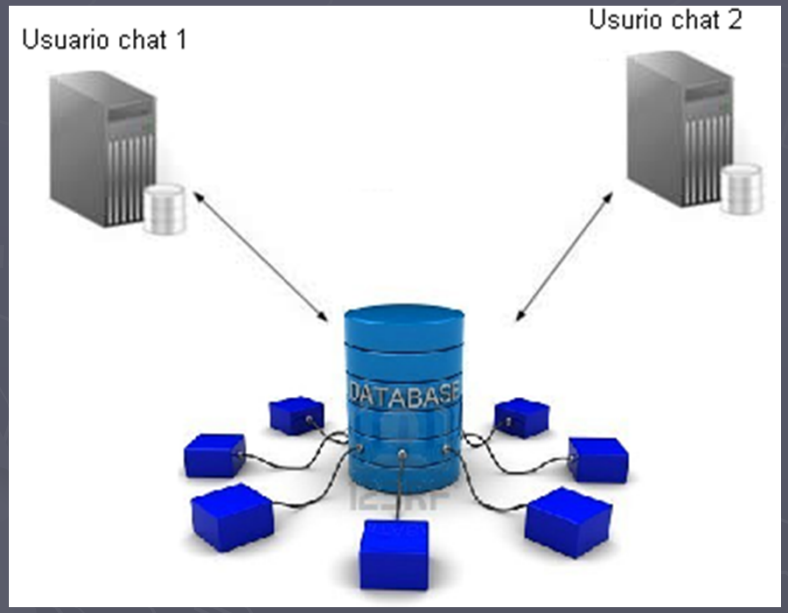
Aplicaciones Multiusuario

- En esta última aplicación se englobarán todos los conocimientos
- Se creará una aplicación multiusuario: un CHAT
- Contendrá una ventana de registro y otra de Chat en tiempo real.

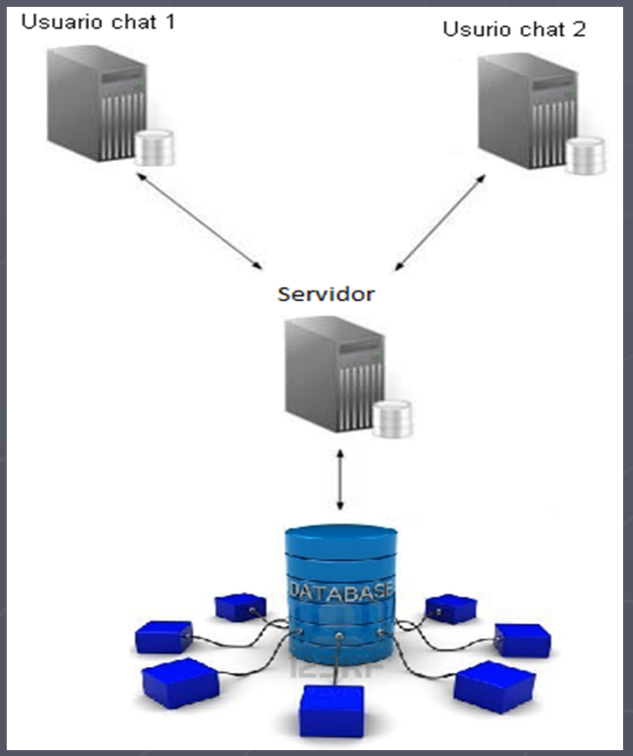
- Director
  - Peticiones constantes a la base de datos en busca de nuevas entradas
- Flash
  - La aplicación cliente realizará peticiones POST constantes al servidor

Aplicaciones Multiusuario: CHAT

- Director



- Flash



Aplicaciones Multiusuario: CHAT

- Director
  - Empleo del API del Xtra XMySQL
  - Realizar peticiones al servidor MySQL mediante XMySQL
  - Control de status del servidor MySQL
- Flash
  - Montar servidor WAMP
  - Empleo del objeto LoadVars para realizar peticiones POST al servidor
  - Generar scripts PHP que ejecuten querys SQL sobre el servidor MySQL y devuelvan resultado obtenido

Aplicaciones Multiusuario

- Director
  - Requiere un conocimiento más profundo del lenguaje Lingo
  - Ideal para aplicaciones de escritorio
  - Requiere el mantenimiento del servidor MySQL
- Flash
  - Requiere conocimientos avanzados en Action Script y PHP
  - En caso de no disponer de un servidor, es necesario montar uno.
  - Independencia entre cliente y servidor

Conclusiones

Conclusiones

- Director
  - Una herramienta cada vez más obsoleta
  - Requiere funcionalidades actuales de las que no dispone (Acceso a BD..)
  - Aplicaciones ocupan mucho espacio en disco, por lo que es poco ideal para la red
- Flash
  - Se ha ido actualizando con el tiempo (AS, AS2, AS3)
  - Las aplicaciones se adaptan a la red
  - Se ha extendido por todo el mundo, y se emplea para un sin fin de aplicaciones